THE HYBRID SIMULATION OF A TACTICAL WEAPON DISCRETE FILTER-CONTROLLER

HENRY GEORGE BOUCHIER HALLAS

THE HYBRID SIMULATION OF A TACTICAL WEAPON DISCRETE FILTER-CONTROLLER

by

Henry George Bouchier Hallas
Lieutenant, Royal Canadian Navy
B.A.Sc., University of Toronto, 1959

[redacted]

Submitted in partial fulfillment
for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

UNITED STATES NAVAL POSTGRADUATE SCHOOL
May 1966

ABSTRACT

This hybrid simulation demonstrates the feasibility of using a digital computer for the realization of a discrete filter-controller to drive a 3"/50 gun mount in response to step, ramp and stabilization orders. The gun position designation signals are subject to random environmental noise and the observation of position is contaminated by random measurement noise. These noise sources are assumed to exhibit a normal distribution with zero mean. A filter is used to predict plant position and velocity from a noisy observable position, and a controller is used to provide the desired feedback of these states for a ripple-free response. The simulation results indicate that there is no detectable difference in response between constant filter gains and adaptive filter gains for this time-invariant plant. The results also demonstrate that the application of good programming techniques is paramount in minimizing the timing and scaling limitations imposed by hybridization.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

LIST OF SYMBOLS

Symbol	
ADC	Analog-to-digital converter
DAC	Digital-to-analog converter
A/D	Analog-to-digital conversion process
D/A	Digital-to-analog conversion process
$(xxxx)_8$	xxxx is a number expressed in the octal number system

1. Introduction

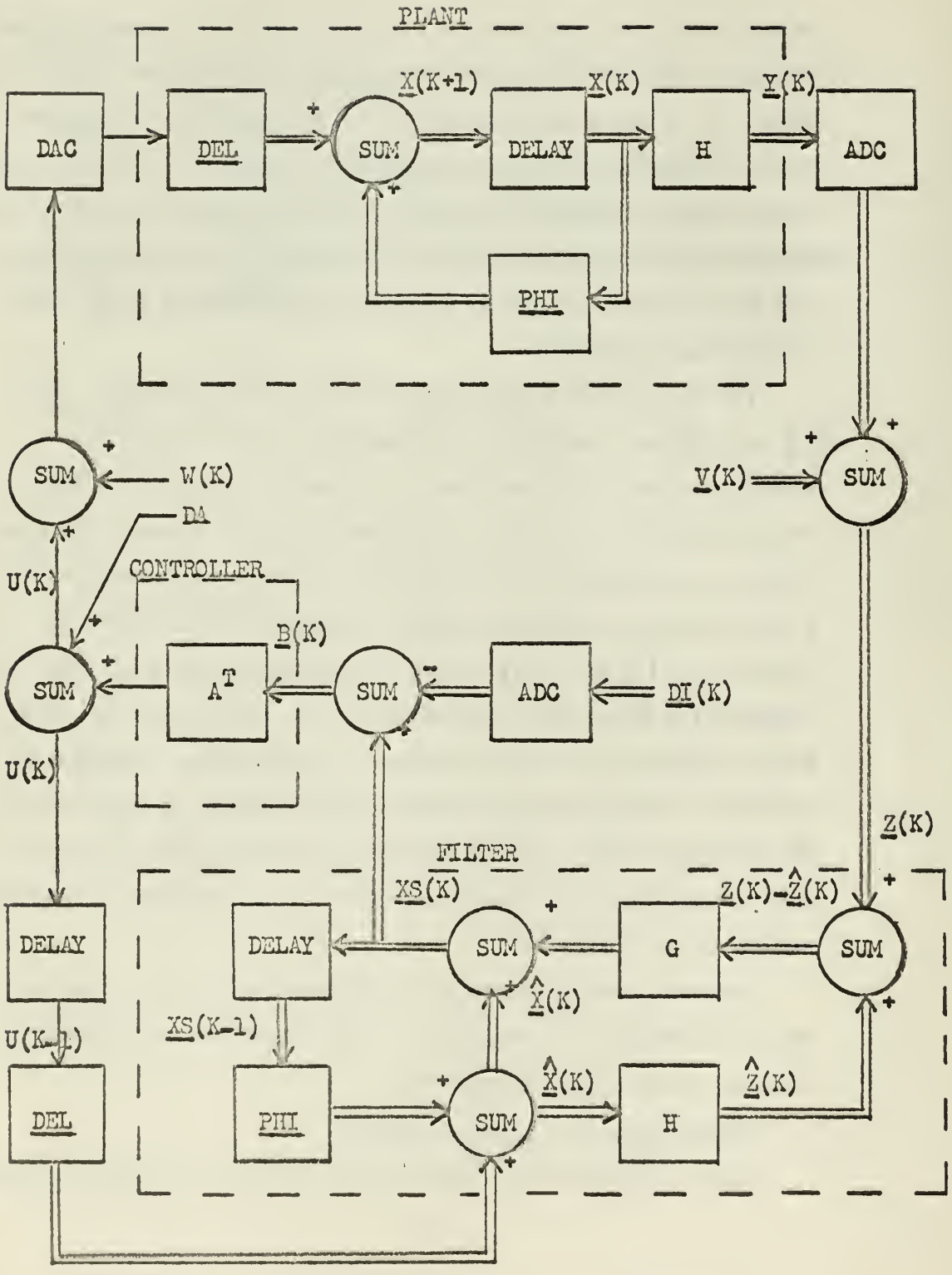
In the past few years a great deal of progress has been made in the theoretical application of state space techniques for the design of sampled-data control systems. The objectives of this paper are to apply such theory to the design of an optimal controller and Kalman filter for the control of the 3"/50 RF Twin Gun Mount and to construct a hybrid simulation of the system so designed, simulating the plant on the Pace TR20 analog computer and representing the filter-controller by a program in the CDC 160 digital computer.

The analytical methods outlined in References [1], [2], [3] and [4] are applied to the design of a discrete filter-controller with constant filter and controller gains to replace the conventional continuous feedback control scheme. These references assume that the system plant may be expressed as a set of linear, constant coefficient, differential equations. References [1] and [2] outline digital computer programs that compute the gain matrix elements for the filter, and the feedback coefficients of the controller, respectively. Reference [4] presents a very thorough summary of the theory as applied to the optimum filter-controller problem and introduces the treatment of deterministic inputs as shown in the general simulation schematic of Figure 1-1.

System operation with constant filter gains is compared to that with adaptive filter gains, where the latter are calculated on-line in the digital computer.

2. Description of 3"/50 Gun Mount

The conventional gun control system uses a typical synchro-



FILTER CONTROLLER SIMULATION SCHEMATIC

FIGURE 1-1

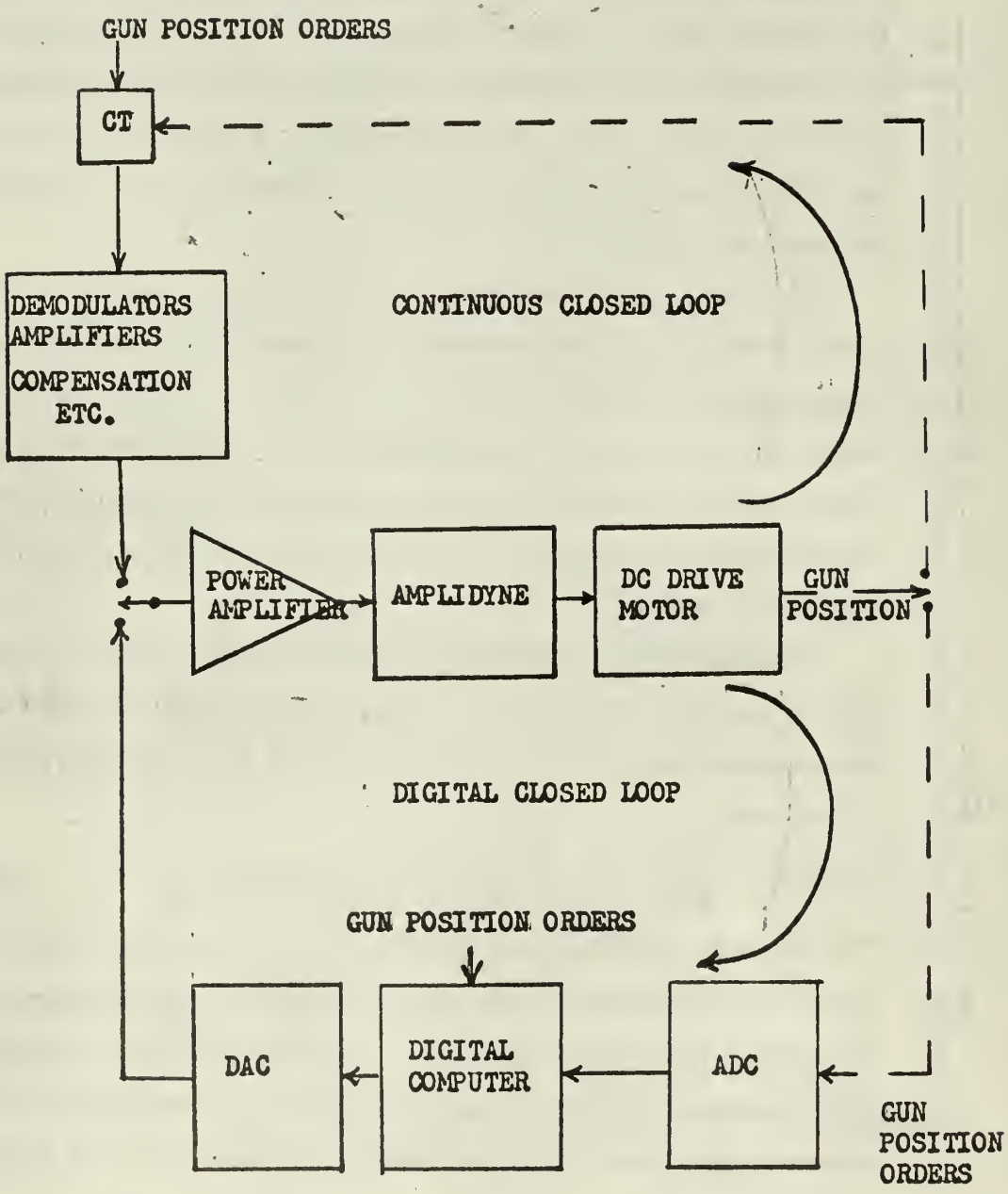
amplidyne drive system as shown in Figure 2-1. Position error is determined by a control transformer, the error voltage is then demodulated, amplified, compensated and applied to the amplidyne power amplifier. The train and elevation servos are essentially identical. The system has a coarse control synchro for slewing onto the target bearing quickly and a fine control synchro for tracking the target smoothly. The control performance constraints are a maximum angular velocity of 30 degrees per second and a maximum angular acceleration of 75 degrees per second squared.

To introduce the discrete filter-controller loop, the conventional control loop was broken at the input to the amplidyne power amplifier and the output shaft of the d-c drive motor as shown in Figure 2-1. The breaking of the system at these points allows either the conventional compensation loop and control or the discrete compensation loop and control to be used by simply throwing a switch.

Reference [5] gives the following mathematical description of the gun mount open-loop transfer function for train and elevation as determined by frequency and transient response techniques:

$$G(s) = \frac{K(s^2 + as + b)}{s(s + p_1)(s + p_2)(s + p_3)(s^2 + cs + d)} \qquad (2\text{-}1)$$

where K is the forward path gain, and the three simple poles are introduced by the characteristics of the d-c amplifier, amplidyne, and the drive motor. The complex zeroes and poles are introduced by the mechanical structure of the gun mount which causes an open loop resonant condition at about 8 cycles per second.



CONVENTIONAL AND DIGITAL CONTROL POWER DRIVES

FIGURE 2-1

This transfer function can be approximated by the following second order plant whose time constant represents the combined characteristics of the drive motor, and the effective friction and inertia at the motor shaft:

$$G(s) = \frac{K}{s(s + a)} \qquad (2\text{-}2)$$

This approximate transfer function models the real system well enough for the purposes of this paper.

In the real gun control system, non-linearities exist because of saturation and because of the limits placed on velocity and acceleration that were designed into this control system for protective measures. To simplify the simulation and to be able to apply linear state space theory the plant is assumed to be linear even though this approximate transfer function of equation 2-2 holds only within certain bounds.

GUN MOUNT PERFORMANCE CHARACTERISTICS

	Train	Elevation
Transfer Function	$G(s) = \frac{3.0}{s(s + 3.5)}$	$G(s) = \frac{6.8}{s(s + 3.5)}$
Uncompensated Zeta	Greater than one	0.668
Natural Frequency	1.732 radians/sec	2.62 radians/sec

TABLE 2-1

Assuming unity feedback the train transfer function would have a closed loop response with a damping ratio slightly

greater than one and a natural frequency of 1.732 radians/second. The system is over-damped and has a very low natural frequency. Consequently the response would be sluggish. In reference [5], personnel at the Naval Electronics Laboratory outline the application of a 'dominant mode' type of compensator which introduces a more desirable pole location having a damping ratio of 0.7 and a natural frequency of ten radians/second. Using a sampling rate of 0.1 seconds and this 'dominant mode' compensator they were able to provide a digital control to the 3"/50 gun mount and to obtain a ripple free response to step and ramp inputs.

3. Gun Position Orders

The gun, however, must in addition respond to sinusoidal stabilization signals of roll and pitch as well as the deterministic inputs of a step and a ramp. Furthermore these signals are all contaminated by noise.

For this paper the sea motions of roll and pitch are defined by the following equations:

$$\text{Roll} = AR*\sin(w_r t) \qquad (3\text{-}1)$$

$$\text{Pitch} = AP*\sin(w_p t) \qquad (3\text{-}2)$$

where

AR is roll amplitude in degrees

AP is pitch amplitude in degrees

T_r is the roll period in seconds, $T_r = 10$ seconds

T_p is the pitch period in seconds, $T_p = 5$ seconds

and the magnitude of the roll amplitude is assumed to be twice that of the pitch for each sea state.

In the hybrid simulation the stabilization signals are sampled and subsequently represented by the output of a zero

order hold. It is assumed that the random noises appearing on the stabilization signals are Gaussian, with zero mean and with standard deviations that can be determined.

In the conventional mode of operation of the gun mount, stabilization signals would be applied continuously. The magnitude and direction of these signals is a function of the sea state. They are required to keep the gun stationary with respect to a horizontal plane. When a target is designated to the gun, a step input causes the gun to slew to this position at maximum velocity. Upon acquiring the target, a ramp input is required for smooth tracking of the target.

It is the intention of this study

1) to use a filter to estimate the plant state variables (gun position and velocity), and

2) to use a controller to provide the right combination of these estimated states in forming a scalar control to obtain an optimum response. The implementation of these ideas, as depicted in Figure 1-1, will be described in the succeeding sections.

4. State Space Description of the Plant

Consider the general second order plant of equation 2-2 to describe the train and elevation motions of the gun. This plant has been assumed to be a linear, time-invariant system and by the methods outlined in Appendix I can be described as a set of first order, linear differential equations given in the following matrix format:

$$\dot{\underline{x}} = F\underline{x} + D u \qquad (4\text{-}1)$$

where and

$$F = \begin{bmatrix} 0 & 1 \\ 0 & -a \end{bmatrix} \qquad D = \begin{bmatrix} 0 \\ k \end{bmatrix}$$

The state transition matrix can be found as a solution of the unforced state equation in the following manner:

$$\dot{\underline{x}} = F\,\underline{x}$$

$$s\,\underline{x}(s) - \underline{x}(0) = F\,\underline{x}(s)$$

$$\underline{x}(s) = \left[sI - F\right]^{-1}\underline{x}(0)$$

where I is the identity matrix

but

$$\left[sI-F\right]^{-1} = \begin{bmatrix} 1/s & 1/s(s+a) \\ 0 & 1/(s+a) \end{bmatrix}$$

For the continuous case,

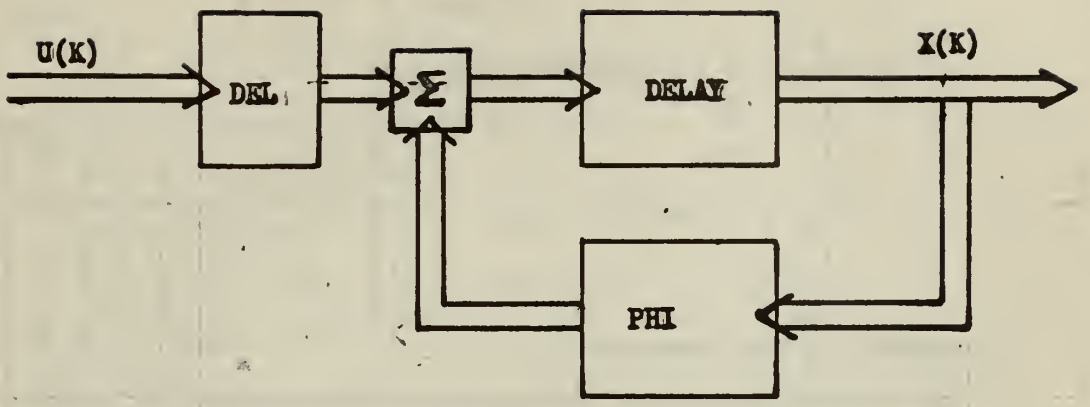
$$\Phi(t-t_o) = L^{-1}\left[sI-F\right]^{-1} = \begin{bmatrix} 1 & (1-\epsilon^{-a(t-t_o)})/a \\ 0 & \epsilon^{-a(t-t_o)} \end{bmatrix} \qquad (4\text{-}2)$$

The state transition matrix for the discrete case is

$$\Phi = \begin{bmatrix} 1 & (1-\epsilon^{-aT})/a \\ 0 & \epsilon^{-aT} \end{bmatrix}_{T,a} \qquad (4\text{-}3)$$

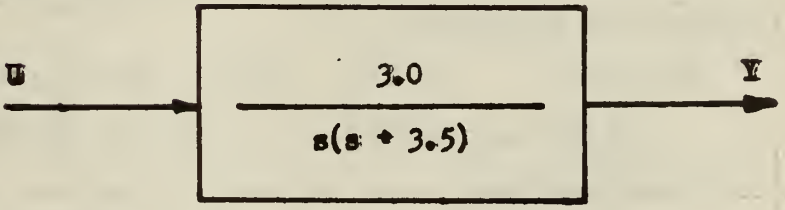
where $T = t-t_o$, and T is the sampling interval and a is the time constant of the plant.

If the control u is represented as the output of a zero order hold and therefore is constant over the sampling interval then the distribution matrix can be found as a solution to the forced state equation for the continuous case in the following manner:



DISCRETE FLOW GRAPH OF THE PLANT

FIGURE 4-1



TRAIN PLANT 3"/50 GUN MOUNT

FIGURE 4-2

$$\Delta = \int_{t_o}^{t} \Phi(t-t_1)\, D\, dt_1$$

$$\Delta = \int_{t_o}^{t} L^{-1} \begin{bmatrix} 1/s & 1/s(s+a) \\ 0 & 1/(s+a) \end{bmatrix} \begin{bmatrix} 0 \\ K \end{bmatrix} dt$$

$$\Delta = \begin{bmatrix} K \int_{t_o}^{t} ((1-\epsilon^{-a(t-t_o)})/a)dt \\ K \int_{t_o}^{t} \epsilon^{-a(t-t_o)}\, dt \end{bmatrix} \qquad (4-4)$$

where K is the plant gain.

The distribution matrix for the discrete case is

$$\Delta = \begin{bmatrix} K \int_{0}^{T} ((1-\epsilon^{-aT})/a)dt \\ K \int_{0}^{T} \epsilon^{-aT}\, dt \end{bmatrix}_{T,a,K} \qquad (4-5)$$

Then the complete discrete state equation depicted graphically in Figure 4-1 is given by

$$\underline{x}(k+1) = \Phi\underline{x}(k) + \Delta u(k) \qquad (4-6)$$

By using the digital program 'Phidel' described in reference [2], the transition and distribution matrices for the train plant shown in Figure 4-2 for a sampling interval of one second were found to be

$$\Phi = \begin{bmatrix} 1.000 & 0.277 \\ 0.000 & 0.030 \end{bmatrix} \quad \text{and} \quad \Delta = \begin{bmatrix} 0.475 \\ 0.651 \end{bmatrix}$$

5. The Controller

It is desired to determine a set of feedback controller gains to control the plant in accordance with a selected performance index. Reference [2] defines one possible performance index

for a discrete sampled-data system as follows:

$$J(N) = \underset{u(k)}{\text{Min}} \sum_{K=M}^{N} \left[x^t(k)Qx(k) + Ru^2(k-1) \right] \qquad (5\text{-}1)$$

where

Q is a nxn positive definite, symmetric, constant matrix

N is number of stages of the process

R is a positive constant

u is the control

x is a nxl vector of plant states

For the minimum time case and for the second order plant being considered in this paper the cost function reduces to

$$J(N) = x_1^2(N) + x_2^2(N) \qquad (5\text{-}2)$$

Reference [2] goes on to develop the recursive algorithms for this discrete final-value control problem as follows:

$$\begin{aligned} \Psi_o &= 0,\ A_o^T = 0,\ P_o = I \\ A_i^T &= \frac{-\Delta^T P_{i-1}\, \Phi}{\Delta^T P_{i-1}\, \Delta} \qquad (5\text{-}3) \\ \Psi_i &= \Phi + \Delta A_i^T \\ P_i &= \Psi_i^T P_{i-1} \Psi_i \end{aligned}$$

where

i = 1 to N; N the number of stages of process for the minimum time case is equal to the order of the system

I is the identity matrix

A_i^T is a lxn vector of feedback controller gains

and

P_i and Ψ_i are matrices involved in the intermediate steps in the recursive solution for the feedback gains.

From the Fortran 60 program 'Optcon', modified to minimize the final states, the following feedback controller gains were obtained for the train plant:

$$\underline{A}^T = \begin{bmatrix} -1.203 & -0.343 \end{bmatrix}$$

A listing of program 'Optcon' is given in Appendix II. Table II-1 gives the definitions for the terms for this program and in addition suggests values of the parameters of the cost function equation 4-1 for the three cases of minimization of time, minimization of control effort, and minimization of both time and control effort.

6. The Filter

In the controller discussion, it is assumed that all the states of the dynamic system are observable states. This condition is certainly not always true. In addition, the observable states may be measured only at the cost of some ambiguity due to measurement noise. The digital filter will provide a best estimate of all the states by weighting the past information with the present observable states. This weighting is performed with the knowledge of the environmental and measurement noise, both of which, it has been assumed, have independent Gaussian distributions with zero mean and a standard deviation that can be determined in some fashion.

The recursive equations 6-1, 6-2 and 6-3 for the Kalman filter are given without proof. References [1] and [3] describe proofs and [4] gives a detailed discussion for a general case. The optimum estimate of the plant states is given by the smoothing equation:

$$\overset{*}{\underline{x}}(k/k) = \hat{\underline{x}}(k/k-1) + G(k)\left[\underline{z}(k) - \hat{\underline{z}}(k/k-1)\right] \qquad (6\text{-}1)$$

where

$$\hat{\underline{x}}(k/k-1) = \Phi \overset{*}{\underline{x}}(k-1/k-1) + \Delta E\left[W(k-1)\right]$$

is the predicted value of the states based upon the last best estimate of the states. The expected value of the environmental noise is zero because it has been assumed to have a normal distribution with zero mean.

$\underline{z}(k) = \underline{y}(k) + \underline{V}(k)$ is the current value of the noisy observable.

$\hat{\underline{z}}(k/k-1) = H\,\hat{\underline{x}}(k/k-1) + E\left[V(k)\right]$ is the predicted noisy observable based on the last value. The expected value of measurement noise is zero because it has been assumed to have a normal distribution with zero mean,

and

$\underline{y}(k) = H\,\underline{x}(k)$ is the current non-noisy value of the observable states.

The filter design problem is based upon the proper selection of the gain matrix, G(k), so that the sum of the variances or errors associated with the estimate of individual states is minimized. The following two recursive relationships are required to calculate the filter gain matrix and the conditional covariance matrix:

$$G(k) = \left[P(k/k-1)H^T(HP(k/k-1)H^T + R)\right]^{-1} \qquad (6\text{-}2)$$

and

$$P(k/k-1) = \Phi\left[P(k-1/k-2) - G(k-1)HP(k-1/k-2)\right]\Phi^T + Q \qquad (6\text{-}3)$$

where

$H = (1\ \ 0)$ is the observability matrix

$R = E\left[\underline{V}^*\underline{V}^T\right]$ is the covariance matrix of measurement noise. Since there is only one observable in the present problem

R is a scalar.

$Q = \Delta E \left[W*W^T \right] \Delta^T$ is the covariance matrix of excitation or signal noise. A discrete schematic of the filter is shown in Figure 1-1.

A stable gain matrix of

$$G = \begin{bmatrix} 0.786 \\ 0.682 \end{bmatrix}$$

was obtained from the Fortran 60 program 'Filter' by providing it with the following parameters:

$$\sigma_w^2 = 0.052$$

$$R(1,1) = 0.02$$

and

$$P(O) = \begin{bmatrix} 0.02 & 0 \\ 0 & 1 \end{bmatrix}$$

A listing of program 'Filter' is given in Appendix II. Table II-2 defines the symbolic terms for this program.

7. The Filter-Controller

The combined filter-controller simulation is shown in Figure 1-1. The double lines represent the transmission of vectors and the single lines, scalars. Deterministic inputs are introduced and are represented by a vector quantity, $\underline{DI}(k)$, with the elements consisting of the input and its successive derivatives. The difference vector, $\underline{B}(k)$, between the deterministic inputs and the predicted states is multiplied by the optimum controller gain matrix, $\underline{A}^T$. A measure of the performance of the system can be deduced from the $\underline{B}$ vector. When the $\underline{B}$ vector is very near zero, the filter is estimating states which are approximately equal to the deterministic inputs.

Then the plant control becomes very small and the system has responded to the inputs.

The output of the controller is a scalar control u(k) which is constant over the sampling interval and is given by

$$u(k) = \underline{A}^T \left[\overset{*}{\underline{x}}(k) - \underline{DI}(k) \right] \qquad (7\text{-}1)$$

This control must be introduced so that it identically affects both the plant and the filter. This is done by feeding the same control to the filter as the plant but delayed by one sampling interval to account for the difference in the sample indicies. The controlled plant and the controlled filter equations are then given by

$$\underline{x}(k+1) = \Phi\, \underline{x}(k) + \Delta\, u(k) + \Delta\, w(k) \qquad (7\text{-}2)$$

and

$$\overset{\wedge}{\underline{x}}(k) = \Phi\, \overset{*}{\underline{x}}(k-1) + \Delta\, u(k-1) \qquad (7\text{-}3)$$

Since the train plant of the gun is a type one system, it has a steady state error in response to a ramp input. This means that the $\underline{B}$ vector would not go to zero unless the $\underline{DI}$ vector is modified. This is accomplished by introducing a DA factor. In this case study, the $\underline{DI}$ vector is made up of two elements:

DI(1,1) = step + ramp*t + stabilization

DI(2,1) = ramp

where

step is the magnitude of the step

ramp is the magnitude of the ramp

and

stabilization is the magnitude of the stabilization signal over one sample interval.

Consider Figure 7-1 and a unit ramp input.

$X_1(00) = t$ $\qquad DI_1 = t$

$X_2(00) = 1$ $\qquad DI_2 = 1$

$\dot{X}_2(00) = 0$

$U(00) = 0$

but

$$\dot{X}_2(00) = 3U(00) - 3.5X_2(00)$$

$$0 \neq -3.5$$

hence introduce DA factor

$$\dot{X}_2(00) = 3DA - 3.5X_2(00)$$

$$DA = 3.5/3$$

A general expression for the DA factor for any magnitude ramp is given by

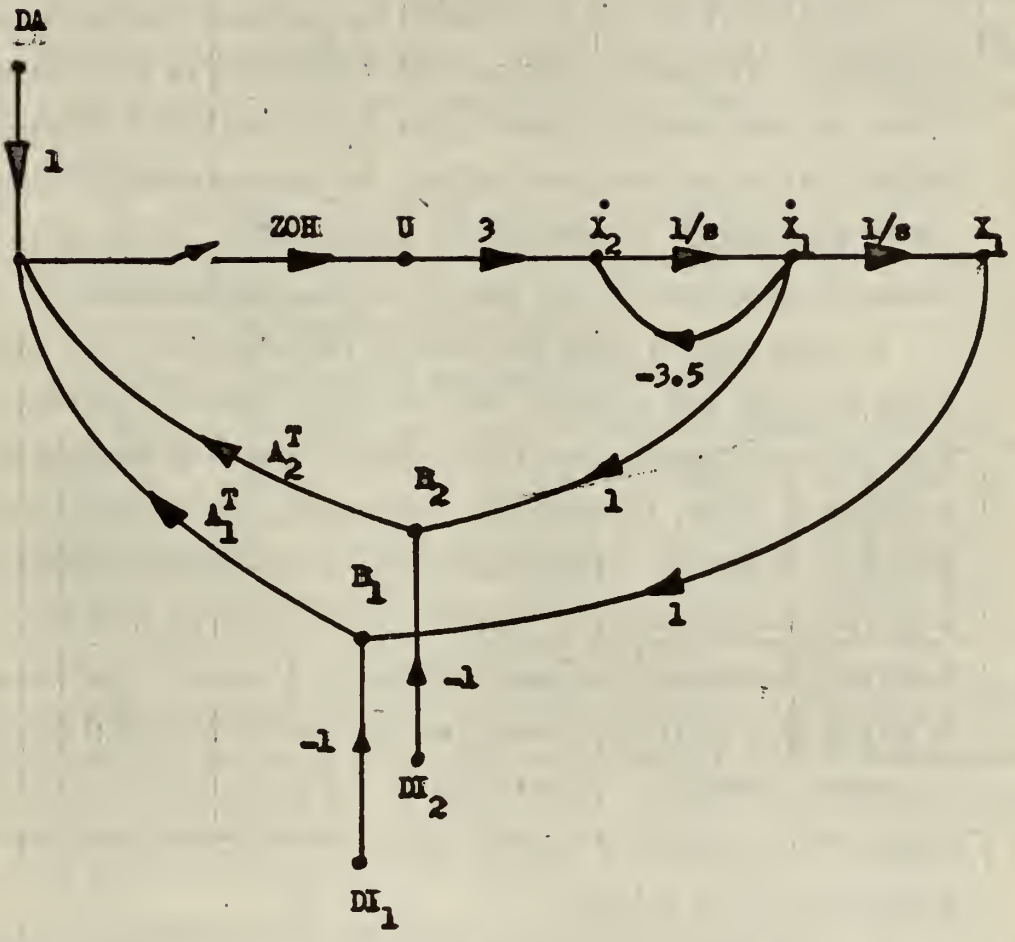
$$DA = \frac{3.5}{3} \text{ ramp} \qquad (7\text{-}4)$$

A Fortran 60 program, 'Hybrid II', was used to simulate the filter-controller design and to compare the results with the hybrid simulation which is described in the following sections. A listing of this program is included in Appendix II.

8. Description of Hybrid Simulation Equipment

The experimental part of this thesis was carried out in the Hybrid Control Laboratory, United States Naval Postgraduate School. Simulation of the system required the following equipment:

A. The Control Data Corporation 160 digital computer is an electronic computer controlled by an internally stored program in sequential locations. Memory capacity is 4096 12-bit words of magnetic core storage, with a storage cycle time of 6.4 microseconds. Instructions are executed in one to four storage or memory cycles with the time required for execution varying



SIGNAL FLOW REPRESENTATION OF PLANT WITH DETERMINISTIC INPUTS

FIGURE 7-1

from 6.4 to 25.6 microseconds or approximately 15 microseconds on the average. A computer word is made up of 12 binary digits numbered from right to left from 0 to 11. All positive numbers must have a zero in bit 11; all negative numbers, a one in bit 11. Hence positive numbers range from $(0000)_8$ to $(3777)_8$ and negative numbers from $(7777)_8$ to $(4000)_8$.

The CDC 160 is very limited in its arithmetic computation capability. Twelve bit addition and subtraction is accomplished in two or three memory cycles using ones complement notation. Multiplication and division can only be accomplished by successive 12 bit addition or subtraction, respectively, resulting in excessive programming and execution time requirements.

B. The Control Data Corporation 168 arithmetic unit provides the CDC 160 computer with single or double precision, fixed point arithmetic operations, either integer or fraction but not floating point, for addition, subtraction, multiplication and division. However, library subroutine mulop provides single precision multiplication in floating point format by first multiplying the numbers and then dividing by a scale factor which in effect post-shifts the answer so that the binary point is in the correct position. Unfortunately, the execution time for mulop is very long; 400 microseconds for no scaling and 550 microseconds for scaling.

C. The Pace TR20 analog computer is a solid state computer having 20 operational amplifiers with a saturation voltage of plus or minus ten volts. By the connection of the operate-reset relays of the TR20 to the same relays of the ADC-DAC, remote control of the analog computer is possible. The run key of the CDC 160 starts the digital program and operates the analog

computer; the master clear key resets the analog computer.

D. The ADC-DAC is a 12 bit conversion unit that functions in the voltage range of zero to minus ten and has a common minus five volts bias source. There are 12 channels analog-to-digital and four channels digital-to-analog. Approximately 100 microseconds is required for each A/D number conversion and approximately 20 microseconds for each D/A number conversion. The input disable jack allows external timing control of the ADC channels.

E. The EAI 1110 Variplotter is a solid state x-y pen recorder with a sensitivity of 100 microvolts per inch. It was used to record the simulation results.

9. Signal Conditioning and Number Scaling

Since the voltage ranges of the analog computer and the ADC-DAC, and the number range of the CDC 160 computer are not compatible, a certain amount of signal conditioning and number scaling is required. Table 9-1 shows the ranges for each piece of equipment and their equivalent values. An $Ax + B$ transformation on the analog voltages is performed to get them into the range of the ADC and thence into the digital computer. A reciprocal transformation is performed for the transfer in the other direction. The A part of the transformation is a scale factor equal to 5/8 for the output analog voltages and equal to 8/5 for the input analog voltages. This division of the output analog voltages by 8 or multiplication by 2^{-3} provides a shifting of the binary point 3 places to the right. Hence all voltages being transferred to the CDC 160 computer have a scale factor of three binary. The B part of the transformation is a five volt bias factor required to convert all analog voltages

COMPARISON OF EQUIPMENT RANGES AND EQUIVALENT VALUES

ANALOG VOLTAGE (DECIMAL) X	SCALED ANALOG VOLTAGE .625X	ADC OUTPUT -(0.625X+Bias)	CORRECT DI GITAL VALUE (OCTAL)	OBSERVED DIGITAL VALUE (OCTAL)
-8	-5.000	-0.000	4000	4003
-7	-4.375	-0.625	4377	4407
-6	-3.75	-1.250	4777	5007
-5	-3.125	-1.875	5377	5363
-4	-2.500	-2.500	5777	5773
-3	-1.875	-3.125	6377	6363
-2	-1.250	-3.750	6777	7003
-1	-0.625	-4.375	7377	7403
0	0.000	-5.000	0000/7777	0003
1	0.625	-5.625	0400	0377
2	1.25	-6.250	1000	0777
3	1.875	-6.875	1400	1377
4	2.500	-7.500	2000	1777
5	3.125	-8.125	2400	2377
6	3.750	-8.750	3000	2777
7	4.375	-9.375	3400	3377
8	5.000	-10.000	3777	3773

TABLE 9-1

NUMBER SCALING

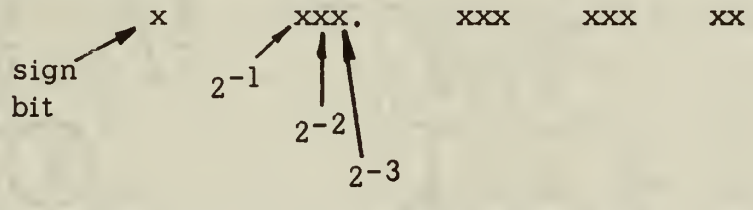
DECIMAL	OCTAL	BINARY	SCALED OCTAL COMPUTER WORD
1.000	1.000	s00/100/000/000/0	0400
-1.203	-1.151	s00/100/110/100/1	7313
-.343	-.257	s00/001/010/111/1	7650
.020	.012	s00/000/000/101/0	0005
.004	.002	s00/000/000/001/0	0001

TABLE 9-2

into the range of the ADC.

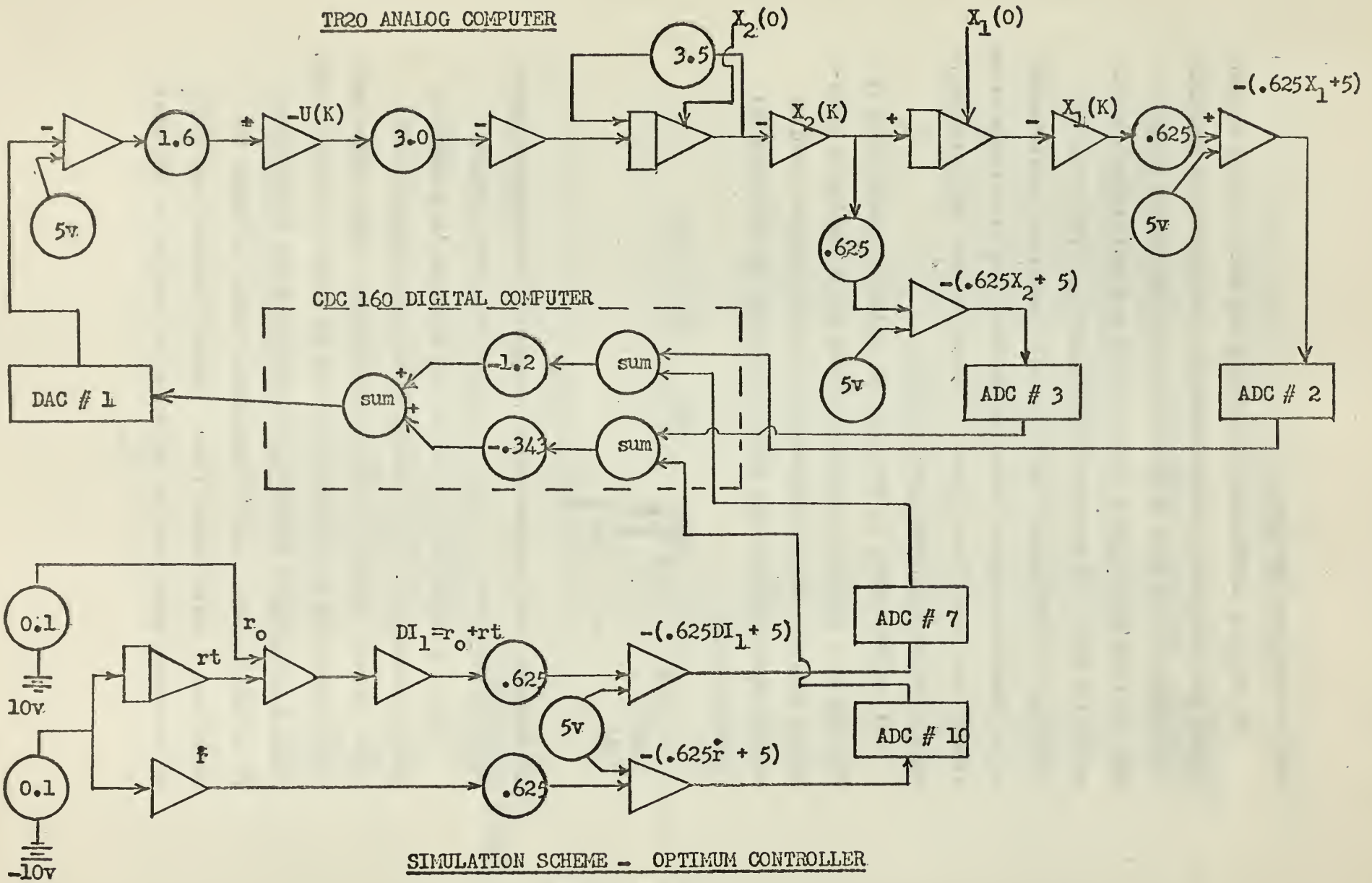
By limiting the analog voltages on the TR20 to plus or minus eight, taking 0.625 of each variable to be outputed to the ADC, and adding it to a bias of plus five volts, the output of the summer is in the range of zero to minus ten volts ready for input to the ADC. To go in the other direction, five volts are added to the DAC output and the entire quantity is multiplied by 1.6. Then the output of the amplifier is in the same range of the analog computer. See Figure 11-1 showing this signal transformation for program Optimum Controller.

Since bit number 11 (the 12th bit of the computer word) is the sign bit, then the remaining 11 bits are available to represent a number. If the decimal point of the computer word is scaled three places to the right, this gives the following bit representation:



This implies that numbers in the range of ±7.999 with a binary scale factor of three can be accomodated by the computer.

Since a scale factor of three was selected for numbers in the digital computer, all constants must be changed to the above format. Table 9-2 shows the decimal, octal, binary and scaled computer word representations for a few constants. Subroutine mulop has a second argument called the scale factor. In multiplying two numbers with scale factors of three, the result has a scale factor of six. Hence a three bit left shift of the binary point must be performed by dividing by the scale



SIMULATION SCHEME - OPTIMUM CONTROLLER

FIGURE II-1

factor.

Since the sign bit must be included one bit of the number is lost and, therefore, the smallest decimal number that can be represented with a binary scale factor of three is .004. However, because of inaccuracies in the ADC-DAC conversion as is shown in Table 9-1, the smallest decimal number used in the hybrid simulation was .02.

10. Arithmetic & Matrix Subroutines

Subroutine mulop was the only available arithmetic subroutine at the start of this study. Therefore, before any simulation could be attempted, basic arithmetic and matrix subroutines had to be written. These subroutines are described in detail in Appendix VI. Error halts are provided in these subroutines for both summation and product overflow. However, no provision is made for over-riding this overflow and using the full positive or negative values. These subroutines are written for single precision or 12 bit numbers. The scaling of the numbers in the computer is flexible and can be changed by modifying all constants entered into the computer. The scale factor in cell $(0027)_8$ must be adjusted, as mentioned in Section 9, to return the binary point to the same place after multiplication or division.

11. Program Optimum Controller

To get a feel for the hybrid simulation problem and to check out the arithmetic and matrix subroutines it was decided to try a simulation that was simple and whose results could be easily verified. Such a simulation problem is the optimal discrete-time control system described by the following equation:

$$u(k) = \underline{A}^T \left[\underline{X}(k) - \underline{DI}(k) \right] \qquad (10\text{-}1)$$

where the true state vector $\underline{X}(k)$ is assumed completely observable without noise contamination. This control law, using feedback coefficients obtained from program 'Optcon' for the minimum time case, will drive the state variables to zero for the regulator problem of initial conditions and to some specific value for the controller problem of deterministic inputs. The system simulation schematic is shown in Figure 11-1. The listing and method of operation of program Optimum Controller are described in Appendix III.

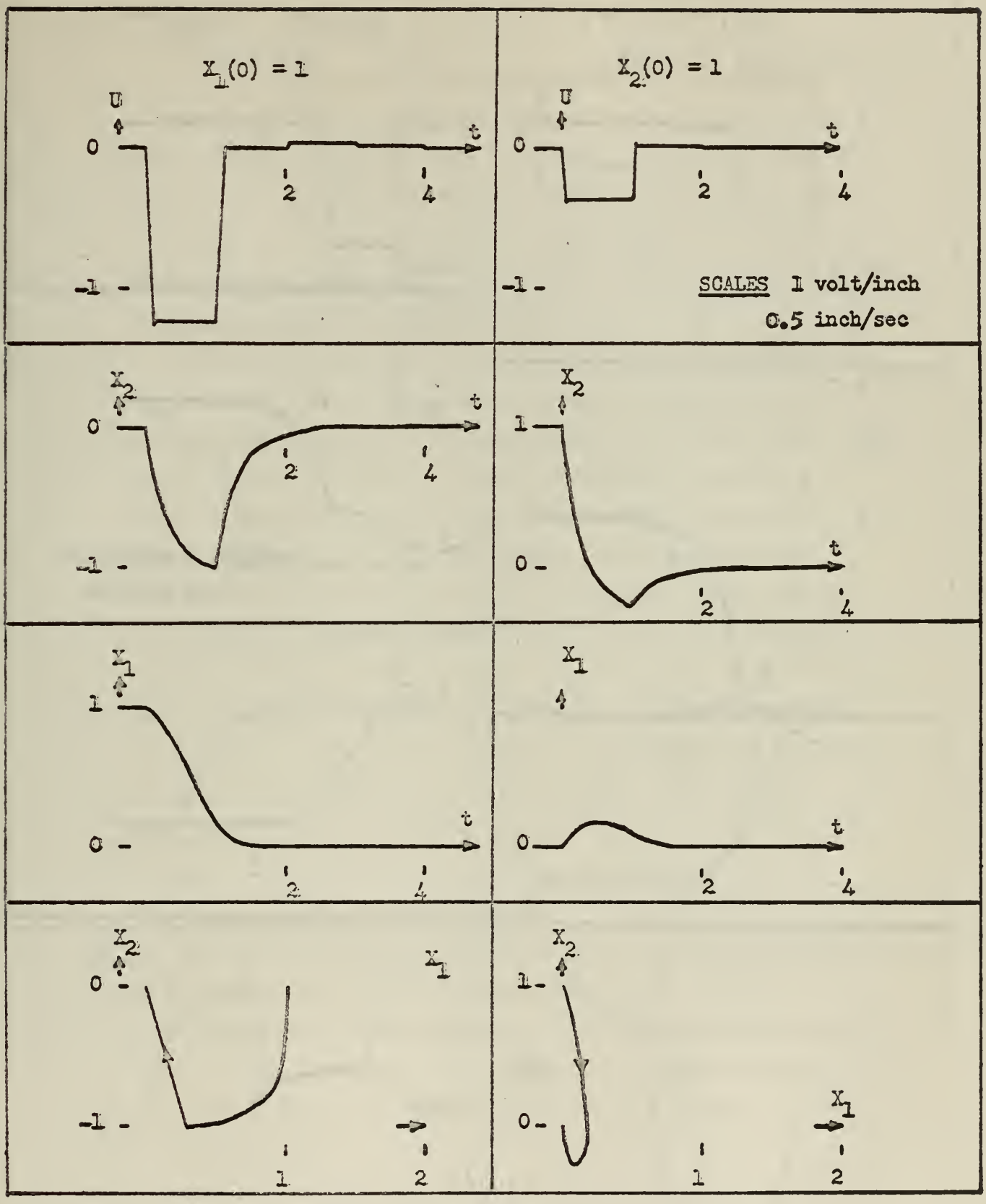
12. Simulation Results for Program Optimum Controller

Figures 12-1 through 12-7 are the x-y recorder plots of the gun train plant response for various initial conditions and deterministic inputs that were limited to the optimal control region of the phase plane. In all cases optimum control is effectively achieved in the minimum time of two samples. The finite response time indicated on the control signal u(t) is due to the inertial lag of the EAI Variplotter. The external timing control was found to be the most accurate, and was used for all recordings. There were no observable hybrid control problems with this relatively simple second order plant and limited inputs. However, for higher order plants, hybrid limitations may be encountered as discussed in the next section.

The approximation of the roll and pitch stabilization signals by the output of a zero order hold proved to be satisfactory. This approximation is used in the filter-controller simulation of Section 18.

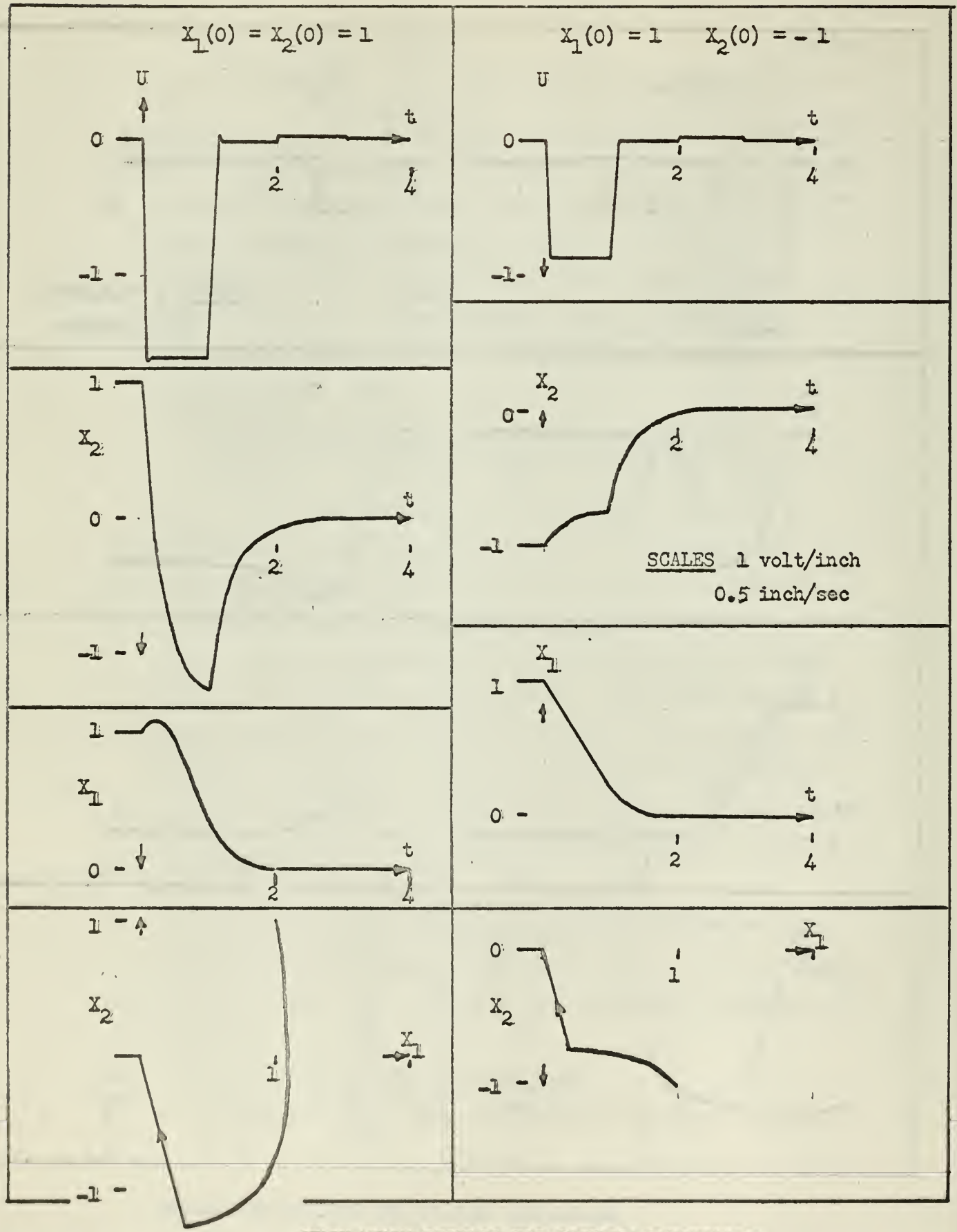
13. Hybrid Control System Limitations

The response of a system controlled by a digital computer



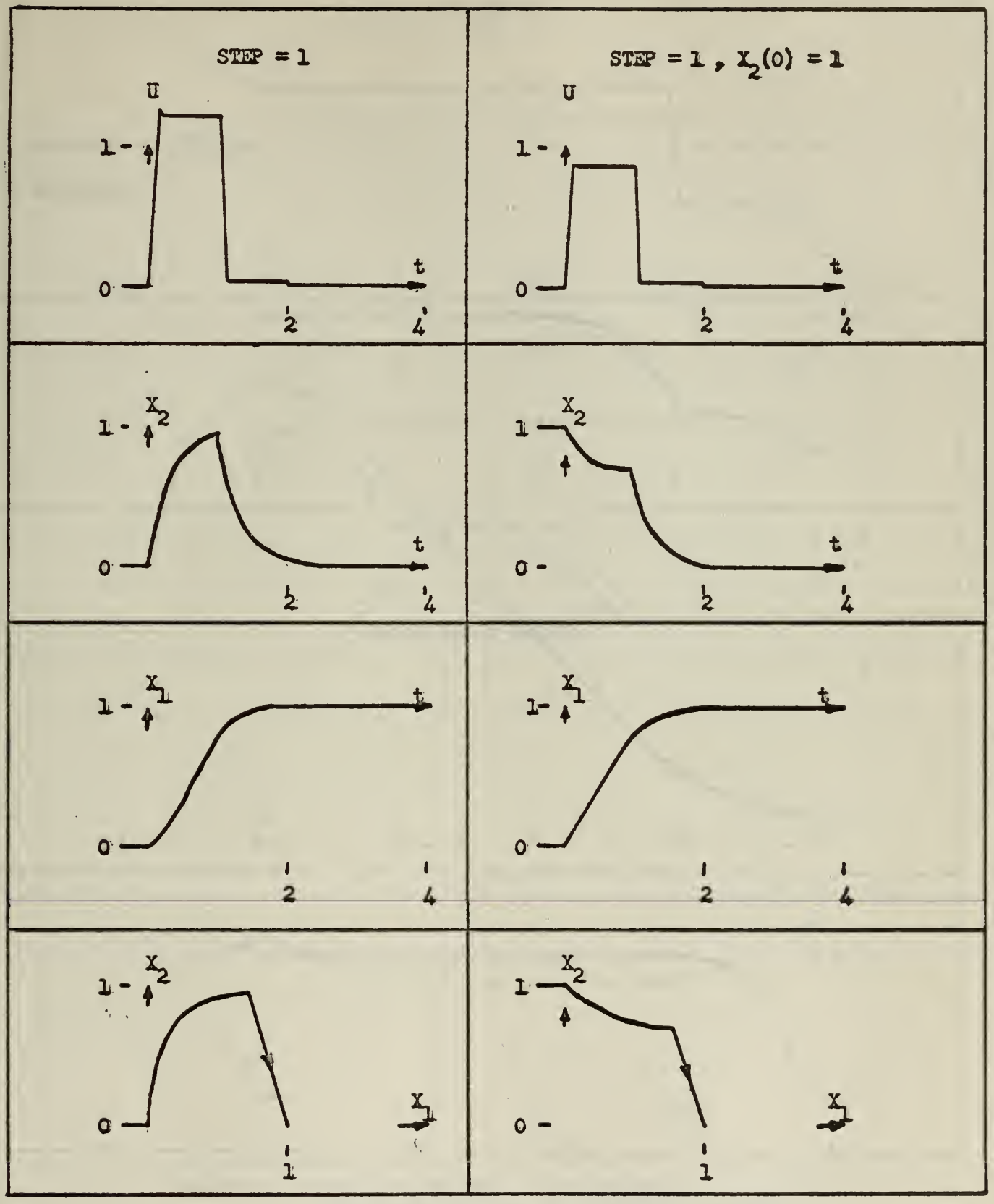
SIMULATION RESULTS FOR OPTIMUM CONTROLLER

FIGURE 12-1



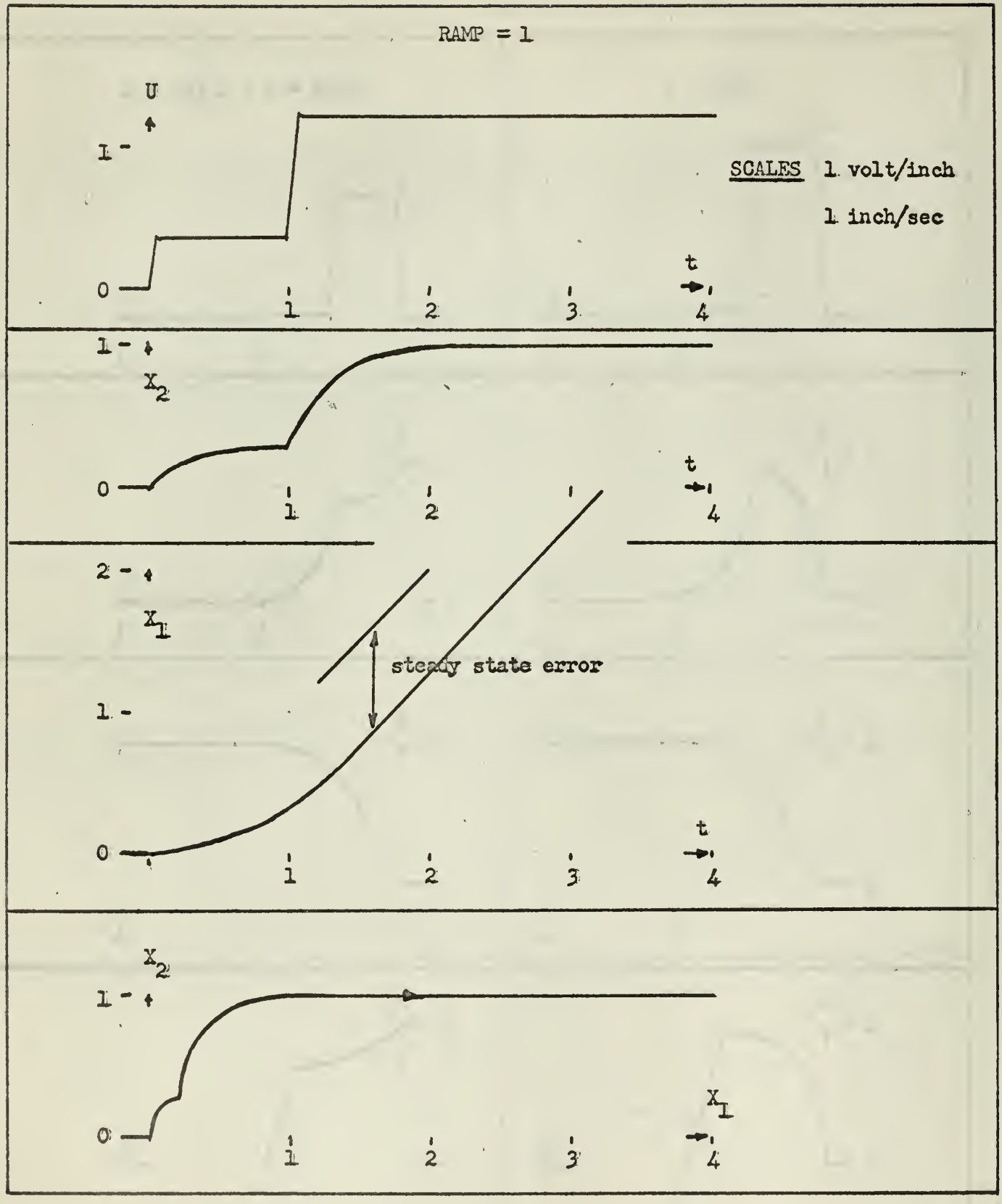
SIMULATION RESULTS FOR OPTIMUM CONTROLLER

FIGURE 12-2



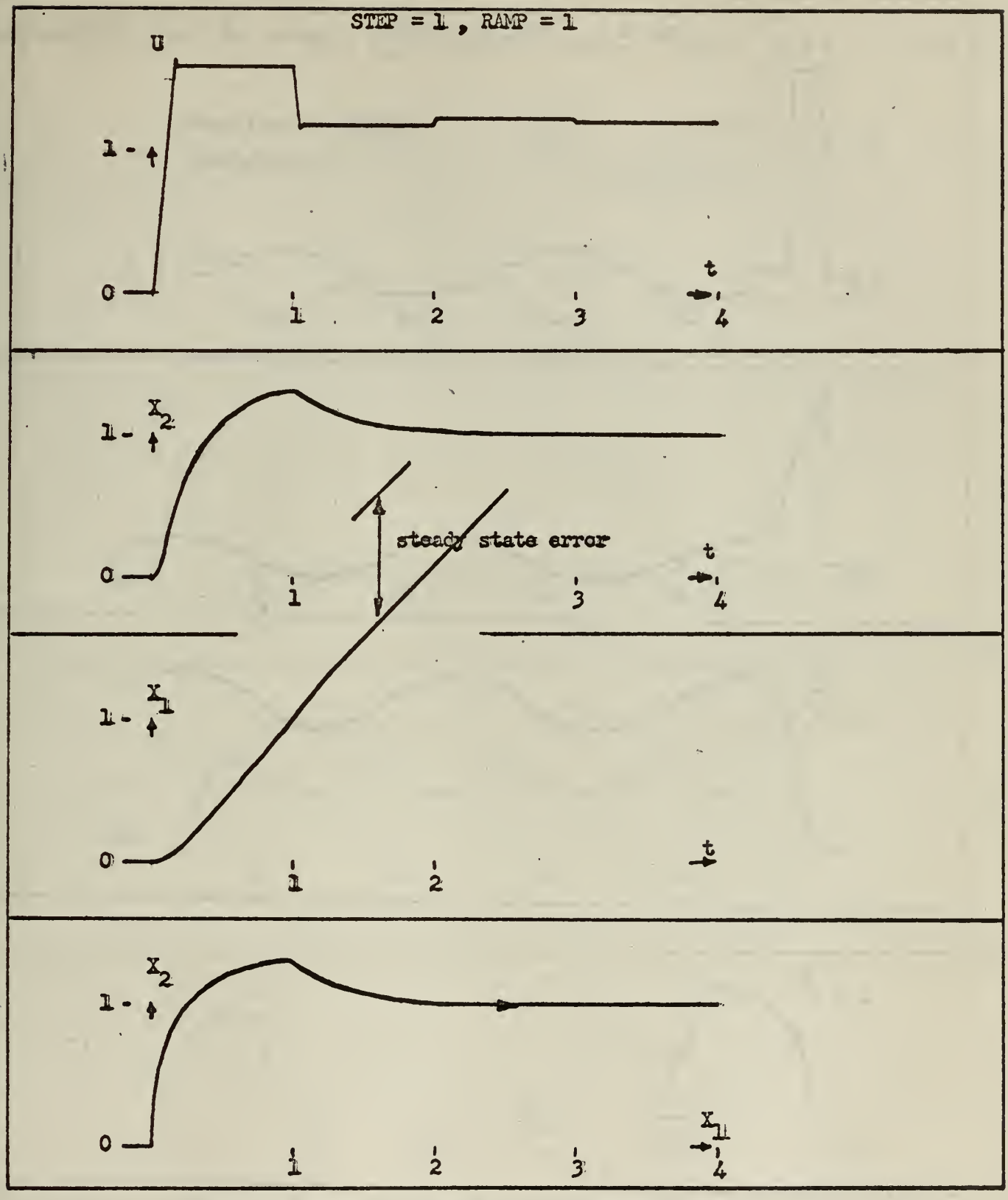
SIMULATION RESULTS FOR OPTIMUM CONTROLLER

FIGURE 12-3



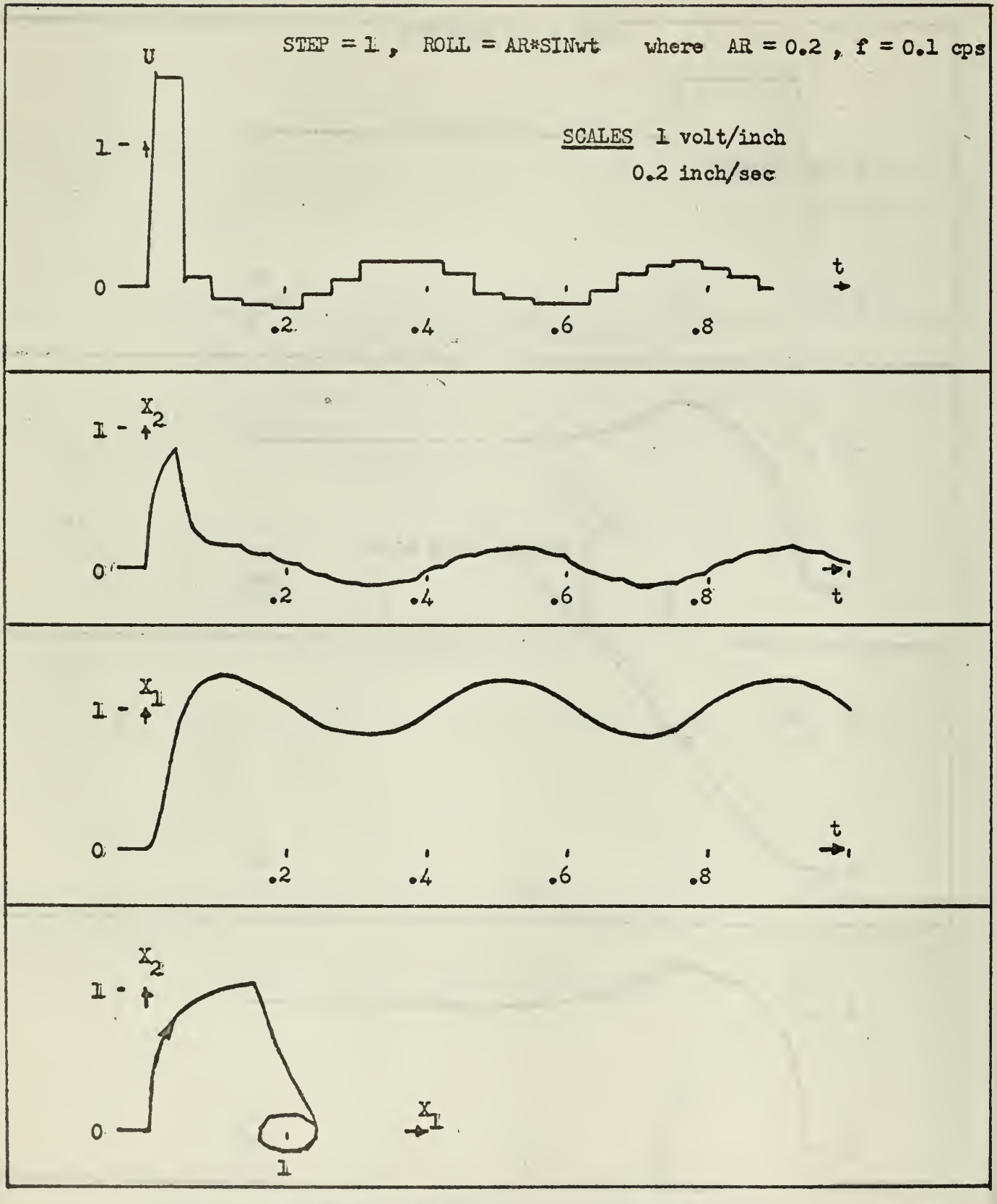
SIMULATION RESULTS FOR OPTIMUM CONTROLLER

FIGURE 12-4



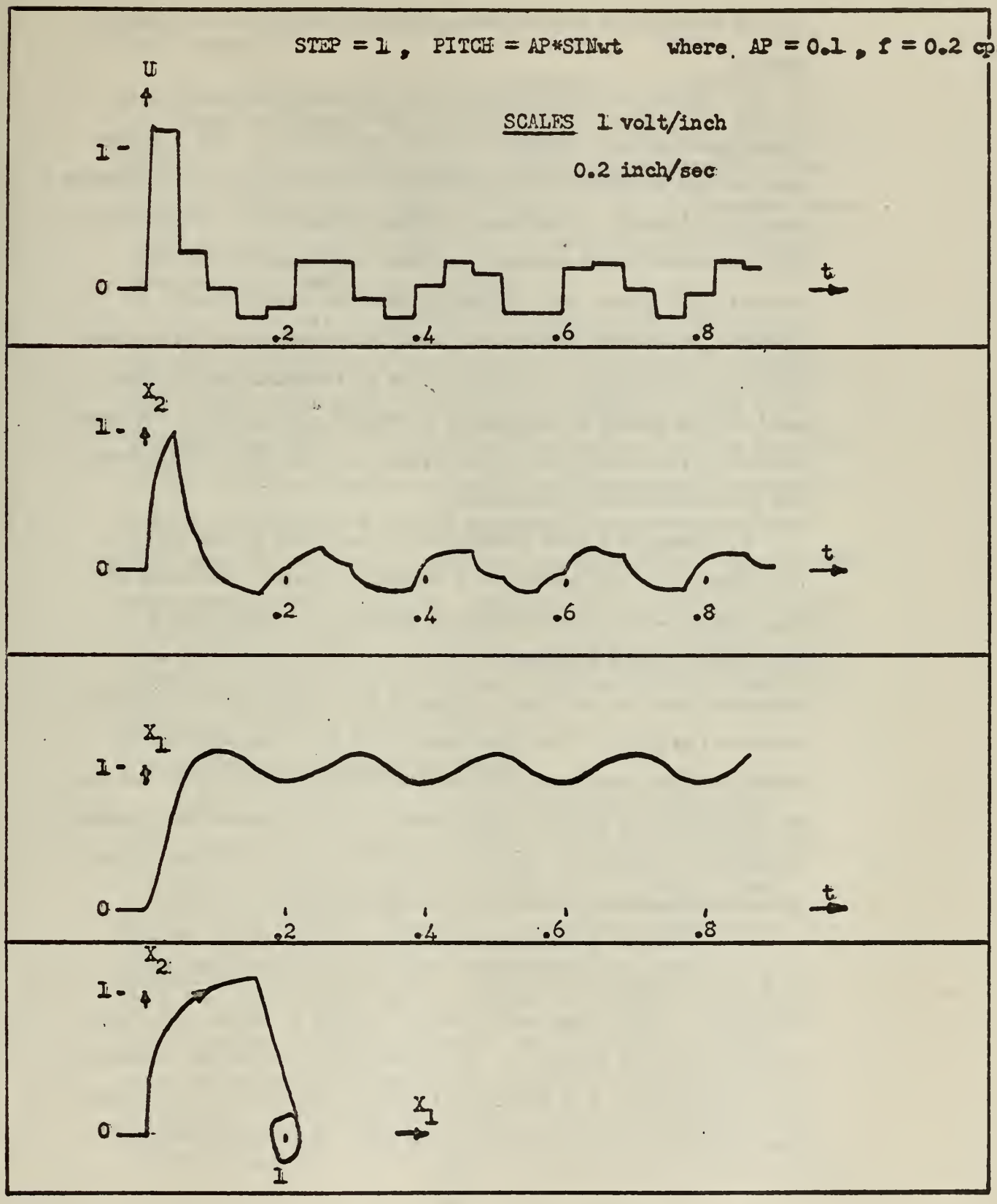
SIMULATION RESULTS FOR OPTIMUM CONTROLLER

FIGURE 12-5



SIMULATION RESULTS FOR OPTIMUM CONTROLLER

FIGURE 12-6



SIMULATION RESULTS FOR OPTIMUM CONTROLLER

FIGURE 12-7

may be affected by the following general limitations of hybridization:

A. There is a time delay due to the serial conversion of analog quantities to digital numbers by the ADC. The seriousness of this problem is determined by the number of simultaneous samples required. In program Optimum Controller, where four A/D conversions per sampling interval are required for this second order plant, this difficulty was not experienced. If, however, the order of the plant was increased, then this effect might be noticeable. In such a case an external sample and hold device would be necessary to sample all inputs at the same instant of time and hold these values until the ADC completes the conversion on all channels.

B. There is a time delay due to the serial nature of the computation of the control in a digital computer. Because of this step-by-step performance of arithmetic operations, a finite time exists between input to the digital computer of samples from the plant and output of the control from the computer to the plant. This time delay is a function of computer speed and the complexity of the computation. This problem can be minimized by using a high speed digital computer with faster arithmetic subroutines than the CDC 160 and by efficient computer programming between the sampling and the output.

C. There is a magnitude limitation imposed by the A/D or D/A conversion processes. This is a physical equipment limitation in this case with a range of plus or minus five volts as discussed in Section 9. In general this magnitude limitation will depend upon the specific ADC-DAC equipment. This problem would be greatly reduced with shaft to digital encoders,

for example.

D. There is a limit in the magnitude of the floating point constants imposed by the computer word length. As discussed in Section 9, the smallest number that can be represented in the computer with a scale factor of three is .004. A computer with an 18 bit word length would allow more flexibility in the computation and entering of constants.

E. There is a sample timing control problem in that the computer is required to sample the inputs at regular intervals. This can be done either internally by subroutine delay or externally by a clock. The internal time delay is not as accurate as the clock because the actual time delay depends upon variations in computation time of arithmetic operations that may vary from cycle to cycle. This fact also makes the internal time delay much harder to adjust. External timing control was used in the hybrid simulation schemes. This type of external clock control is much more accurate and would allow time sharing of the computer with other equipments.

14. Selection of Sampling Interval

In the selection of a sampling interval for the digital filter-controller a number of factors must be considered. The inertia of the gun is large and the system is velocity and acceleration limited. Too small a sampling interval would require controls of such large magnitude in the minimum time case that they may cause the system to saturate and operate non-linearly. On the other hand, for too large a sampling interval the late updating of the plant may be too slow for a fast target. If a number of systems are to be time shared by the same computer, then a long sampling interval would allow

more systems to be controlled by the same computer. The types of input signals must also be considered. For instance, in this case the sampling interval must be small enough in order that the roll and pitch stabilization signals can be well represented by the output of a zero order hold. The limitations of the equipment to be used must also be considered. In this case, the parameters and constants had to be within the number range of the computer. Considering all these factors, a sampling interval of one second was chosen.

15. Selection of Noise Values

The choice of the values of measurement and environmental noise, and the initialization of the error covariance matrix can be by conjecture, actual measurement of the statistics or by equipment limitations. In this simulation, in order to stay within the number range of the computer, the selection was predicated by the latter method. It is assumed that the probability density functions of V(k) and W(k) are normal with zero mean. Both V(k) and W(k) are one dimensional random variables. Hence they are scalars. The noise values for the hybrid simulation were selected as follows:

A. The measurement noise is given by

$$R(1,1) = E\left[V(k)^2\right] = 0.02$$

The standard deviation of the measurement noise is given by

$$\sigma_V = 0.1414$$

If a shaft to digital encoder is used to transform the shaft angle output to a digital number, then the ambiguity of this transformation determines the value of R(1,1).

B. The covariance matrix of random disturbance due to

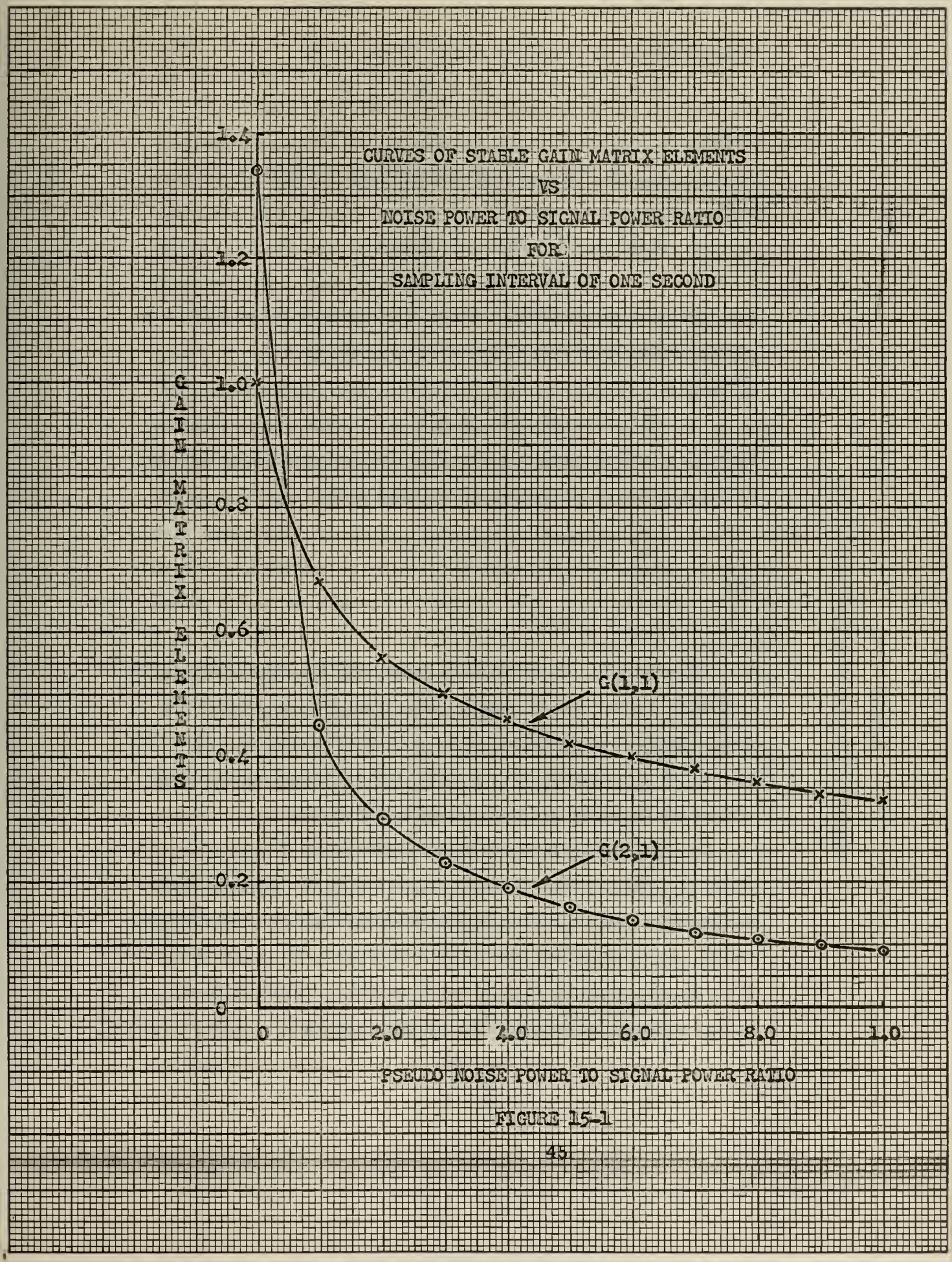

CURVES OF STABLE GAIN MATRIX ELEMENTS
VS
NOISE POWER TO SIGNAL POWER RATIO
FOR
SAMPLING INTERVAL OF ONE SECOND
GAIN MATRIX ELEMENTS
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0
G(1,1)
G(2,1)
0
2.0
4.0
6.0
8.0
1.0
PSEUDO NOISE POWER TO SIGNAL POWER RATIO

FIGURE 15-1

environmental or excitation noise is given by

$$Q = \Delta\, E\left[W(k)\right]\Delta^{t} = \Delta\,\sigma_{w}^{2}\,\Delta^{t}$$

where

σ_{w}^{2} = W(k), is the variance of excitation noise,

and

$$\Delta\,\Delta^{t} = \begin{bmatrix} 0.384 & 0.515 \\ 0.515 & 0.690 \end{bmatrix}$$

The variance of excitation noise is calculated by assuming a pseudo noise to signal ratio of R(1,1)/Q(1,1) = 1.

Therefore

$$Q(1,1) = .02 = \sigma_{w}^{2}\,\Delta\,\Delta^{t}$$

From this relationship the variance and the standard deviation of the excitation noise are calculated to be

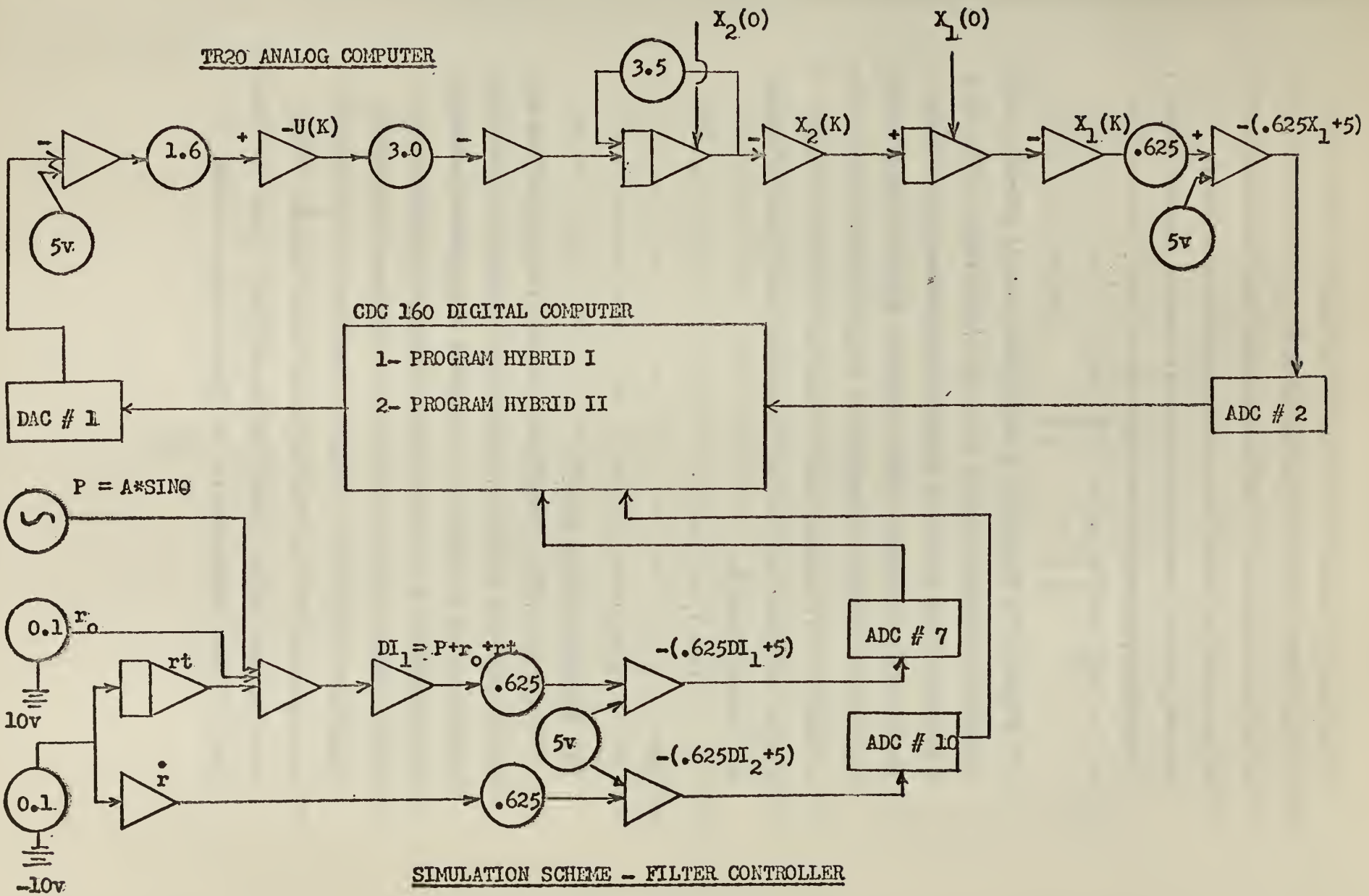
$$\sigma_{w}^{2} = .052$$

and

$$\sigma_{w} = .228$$

For this weapon, the excitation noise can be considered to be a random signal due to jitter. The jitter may be caused by noisy transmission of gun orders, sharp variations in the stabilization or designation signals, and ambiguity in bearing alignment between the designator and the gun.

C. The estimation covariance matrix for the second order plant is a two-by-two matrix. $P_{o}(1,1)$ is the initial variance on position and is based on the statistics of the measurement noise. It is therefore reasonable to assign R(1,1) as the value of $P_{o}(1,1)$. $P_{o}(2,2)$ is a measure of the confidence in the initial velocity measurement. In this simulation, the velocity



SIMULATION SCHEME - FILTER CONTROLLER

FIGURE 16-1

of the plant is unknown and must be estimated by the filter. Consequently, a large value compared to $P_o(1,1)$ is chosen to show this uncertainty. Off-dimensional terms are assigned zero value since the initial state estimates are uncorrelated. The following error covariance matrix was used in program Hybrid II to calculate filter gains on-line:

$$P_o = \begin{bmatrix} 0.02 & 0 \\ 0 & 1 \end{bmatrix}$$

Figure 15-1 represents a normalized plot of the gain matrix elements for various pseudo noise-to-signal ratios. $R(1,1)$ is a measure of the expected position measurement noise power and $Q(1,1)$ is a measure of the expected position signal power. This plot is made by considering a pseudo noise-to-signal ratio in which $R(1,1)$ is made equal to multiple values of $Q(1,1)$. For a noise to signal ratio of one, the stable gain matrix elements are always given by

$$G(1,1) = 0.68$$

and

$$G(2,1) = 0.45$$

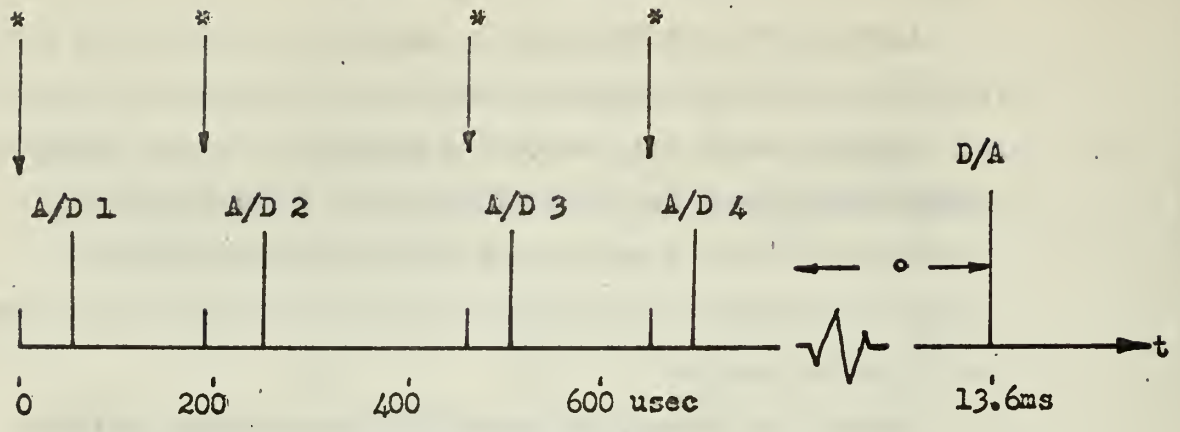
16. Programs Hybrid I & II

Programs Hybrid I and II were written in machine language for the CDC 160 computer to simulate the discrete digital filter-controller part of Figure 1-1. Hybrid I is described thoroughly in Appendix IV, and uses constant filter and controller gains. Hybrid II is described in Appendix V, and uses on-line computation of filter gains. These programs solve the controlled plant equation 7-2 and the controlled filter equation 7-3 to calculate the digital control for the weapon. Figure 16-1 shows the simulation schematic for the hybrid control system.

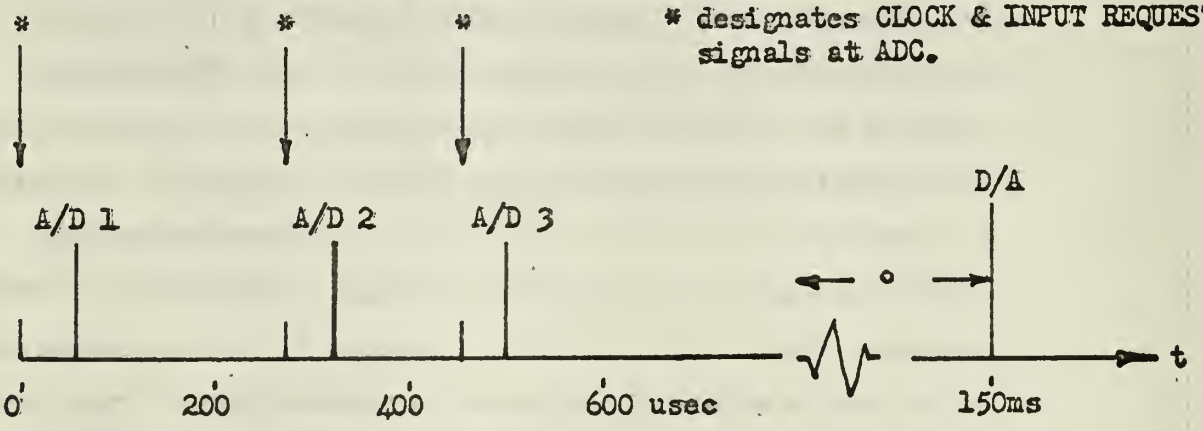
In the writing of the Hybrid programs every effort was made to minimize the hybridization limitations discussed in Section 13. For example, in the first edition of Hybrid II, the time between sample and output was 250 milliseconds, However, by the application of good programming techniques and by the relocation of certain computation segments, the delay was reduced to 130 milliseconds.

Figure 16-2 shows the time history for programs Optimum Controller, Hybrid I and II. In all three programs the A/D sampling takes 820 microseconds or less and therefore this limitation is small compared to the computation delay times. A computation delay of 13.6 milliseconds did not affect the response of the system as observed and discussed in Section 12 for the simulation results of program Optimum Controller. However, a computation delay time of 130 to 150 milliseconds for the Hybrid programs does affect the response of the system as shown in Figures 16-3, 18-1 and 18-2. In these Figures a comparison of the step response of the system is made between normal gun operation being sampled every second, and the system time scaled by a factor of ten being sampled every ten seconds. Figure 16-4 illustrates in more detail what happens when this compute time 't_c' is significant. The time axis of this Figure is expanded. If $t_c = 0$, then the first and succeeding controls are based on the samples of the plant at the proper instant and the controls are applied to the plant at that same instant producing the desired optimum response. However, if $t_c > 0$, then the control is applied t_c seconds after each sample time. In the mean time the plant states are advancing without the proper

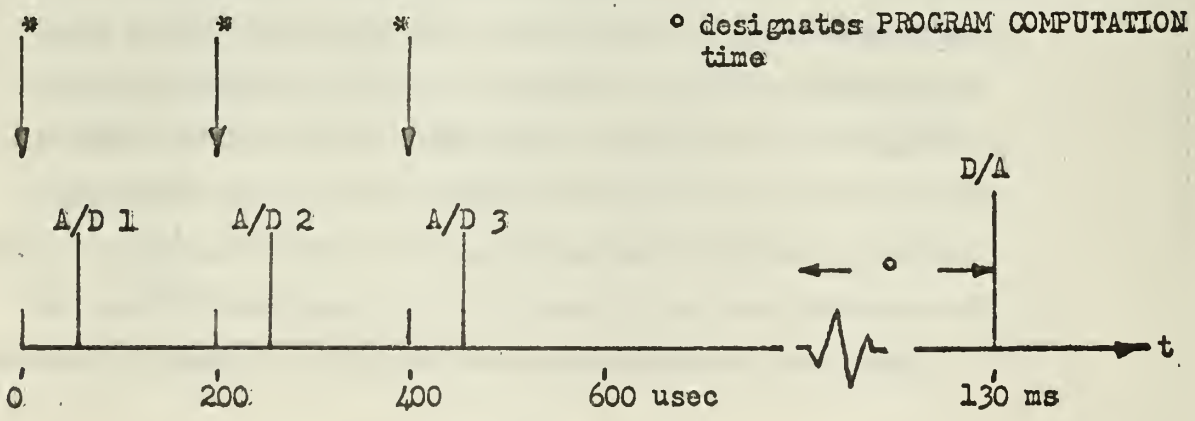
PROGRAM TIME HISTORY GRAPHS



OPTIMUM CONTROLLER

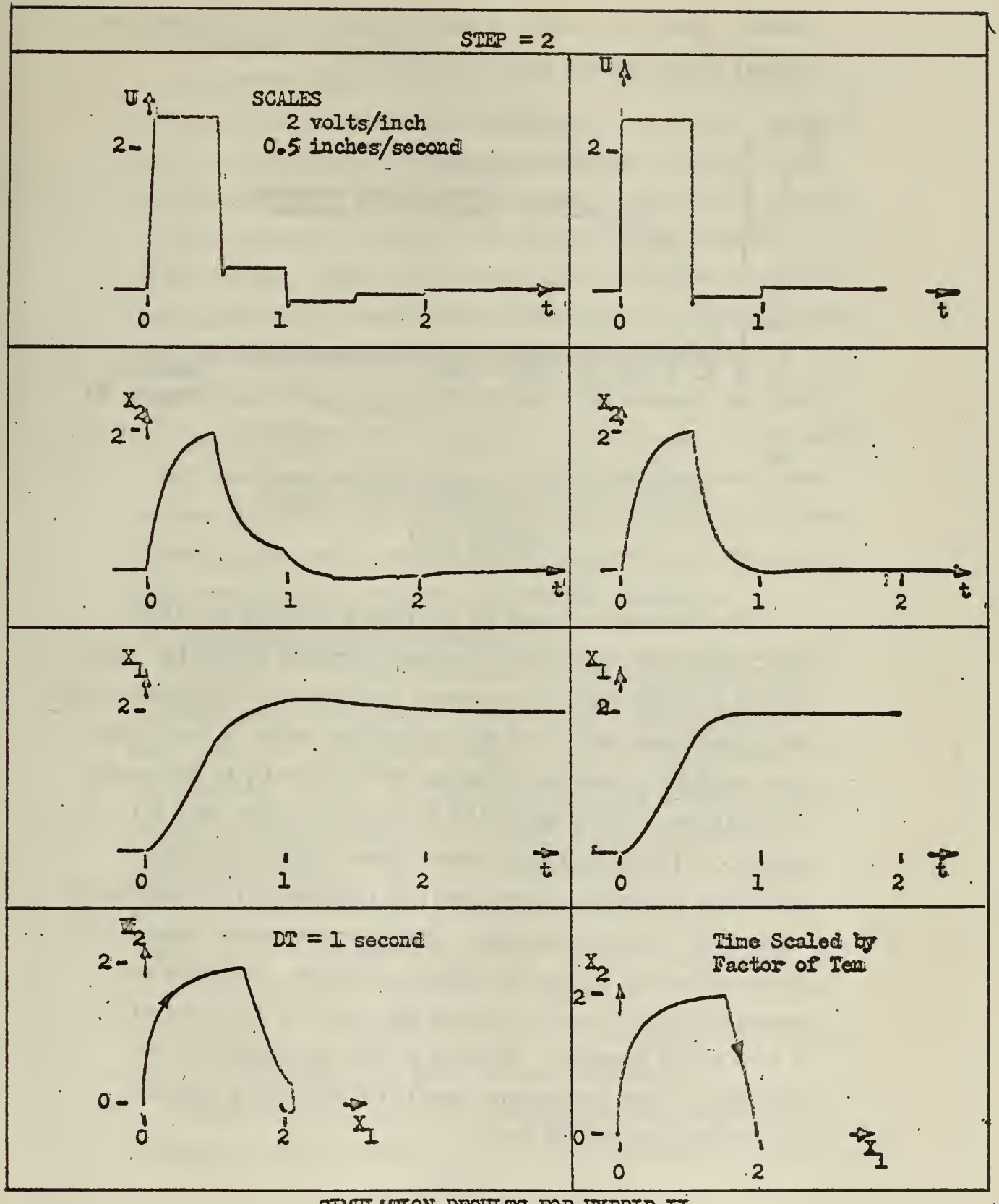


HYBRID I



HYBRID II

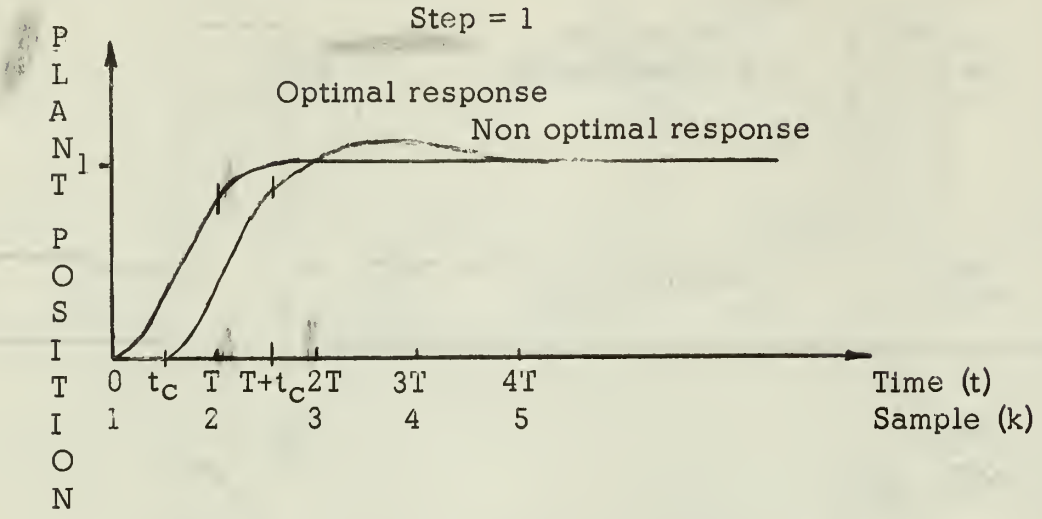
FIGURE 16-2



SIMULATION RESULTS FOR HYBRID II

FIGURE 16-3

control. When the control is finally applied it is no longer the optimal control for the plant at time $KT + t_c$.



EFFECT OF COMPUTATION TIME DELAY

FIGURE 16-4

The computation times for the hybrid programs are about ten times longer than that for program Optimum Controller. The Hybrid programs have an increased complexity of computation but the predominant reason for this longer time delay is that a very slow multiply subroutine is called very frequently in the matrix calculations. Mulop takes 550 microseconds each time it is called. It is felt that with newer, higher speed computers and faster arithmetic subroutines, an improvement in computation time of ten to one is feasible. Therefore the results obtained by time scaling the system by a factor of ten are justifiable representation of the type of control that could be achieved with a high speed computer. Because of this consideration, the remainder of the simulations results for the Hybrid programs are for the time-scaled plant.

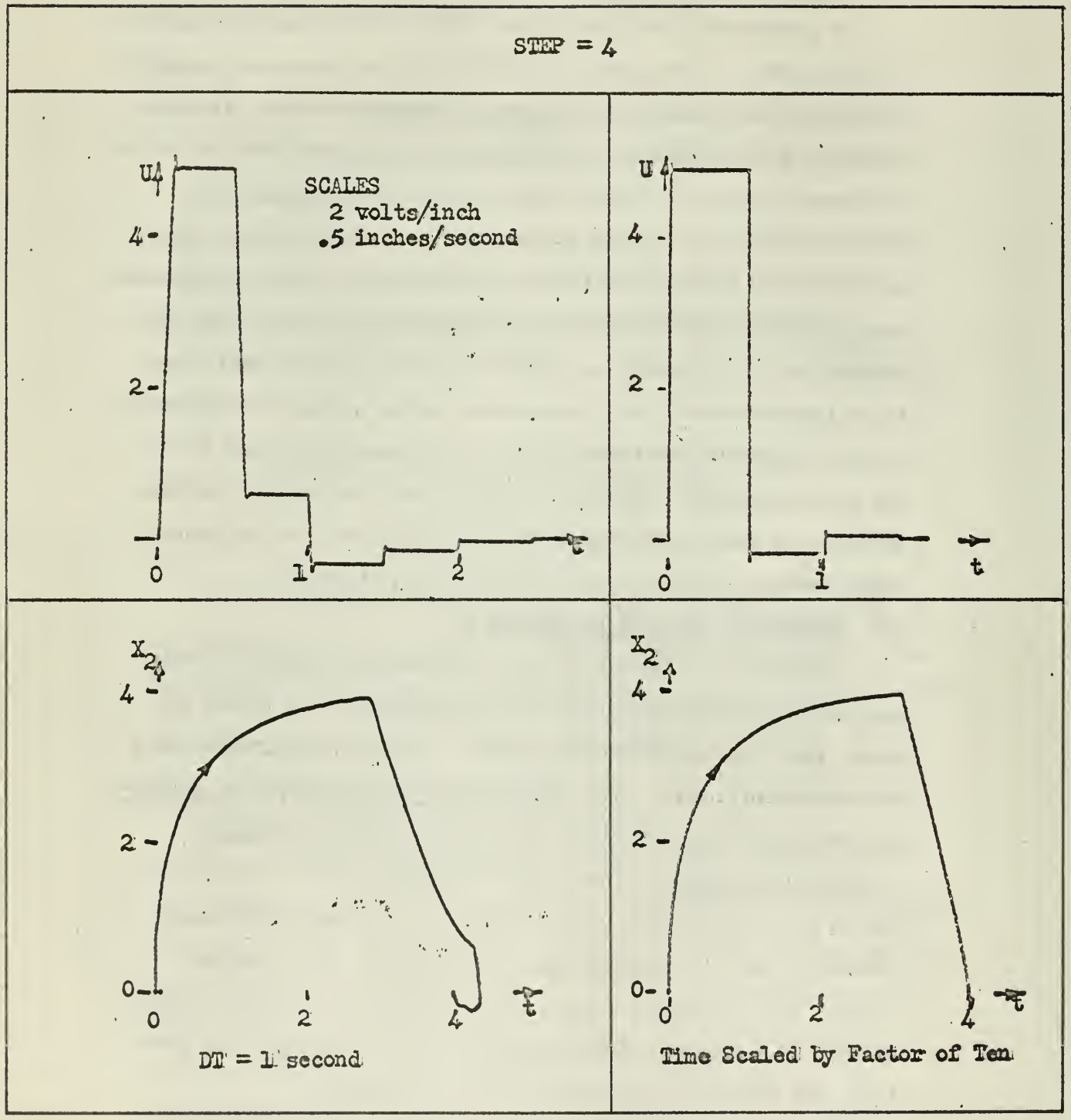
17. Adaptive Filter Gains

In program Hybrid I the steady state filter gains are entered as constants. In program Hybrid II the filter gains are computed on line for each sample by calling subroutine vf gain. In this adaptive filter gain case, the filter gains increase from initially computed values, to steady state values in approximately seven samples. From this instant on, the adaptive filter gains have reached their steady state values and both Hybrid programs have the same response for this time-invariant plant. The simulation results for programs Hybrid I and II indicate that there is no improvement in the response by using adaptive filter gains. In fact duplicate pen recordings of the phase plane were taken for step responses with one response traced on top of the other. Because of this exact reproduction of responses all the simulation results shown are from one program, Hybrid II.

18. Simulation Results for Hybrid II

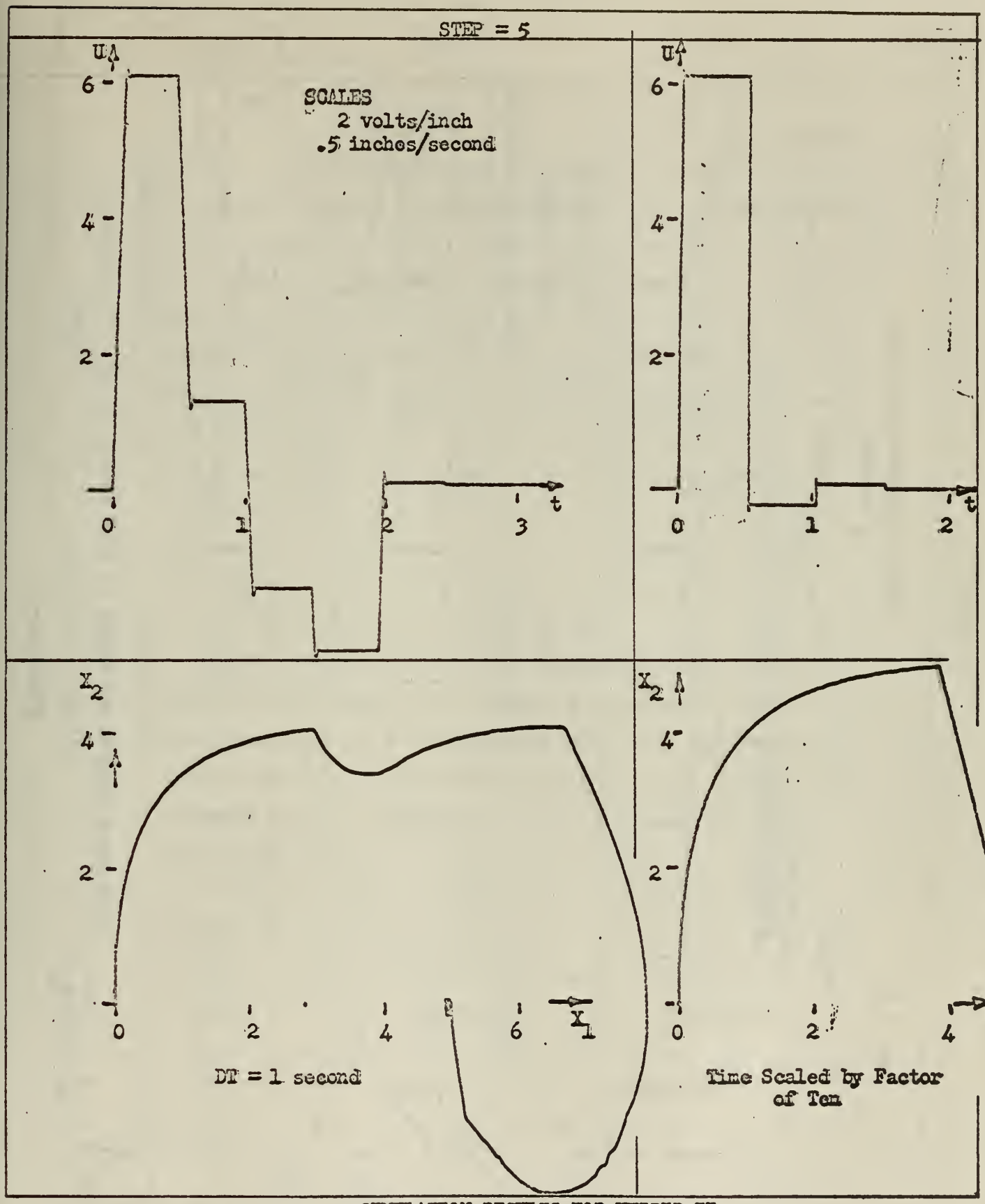
Figures 18-1 through 18-6 are pen recorder graphs of the gun train plant responses for various deterministic inputs of step, ramp and stabilization signals, and for measurement and environmental noise. Two points must be considered in observing the simulation results that contain either noise and/or stabilization signal. The noise is reinitialized to start at the top of the list for each pen recording so that each individual recording can be compared for that particular deterministic input. The sinusoidal stabilization signal is not synchronized to start at the same position for each pen recording, and therefore, the individual time axes are not correlated.

Figures 18-1 and 18-2 illustrate again the hybridization problem that occurs when too long a delay between sample and



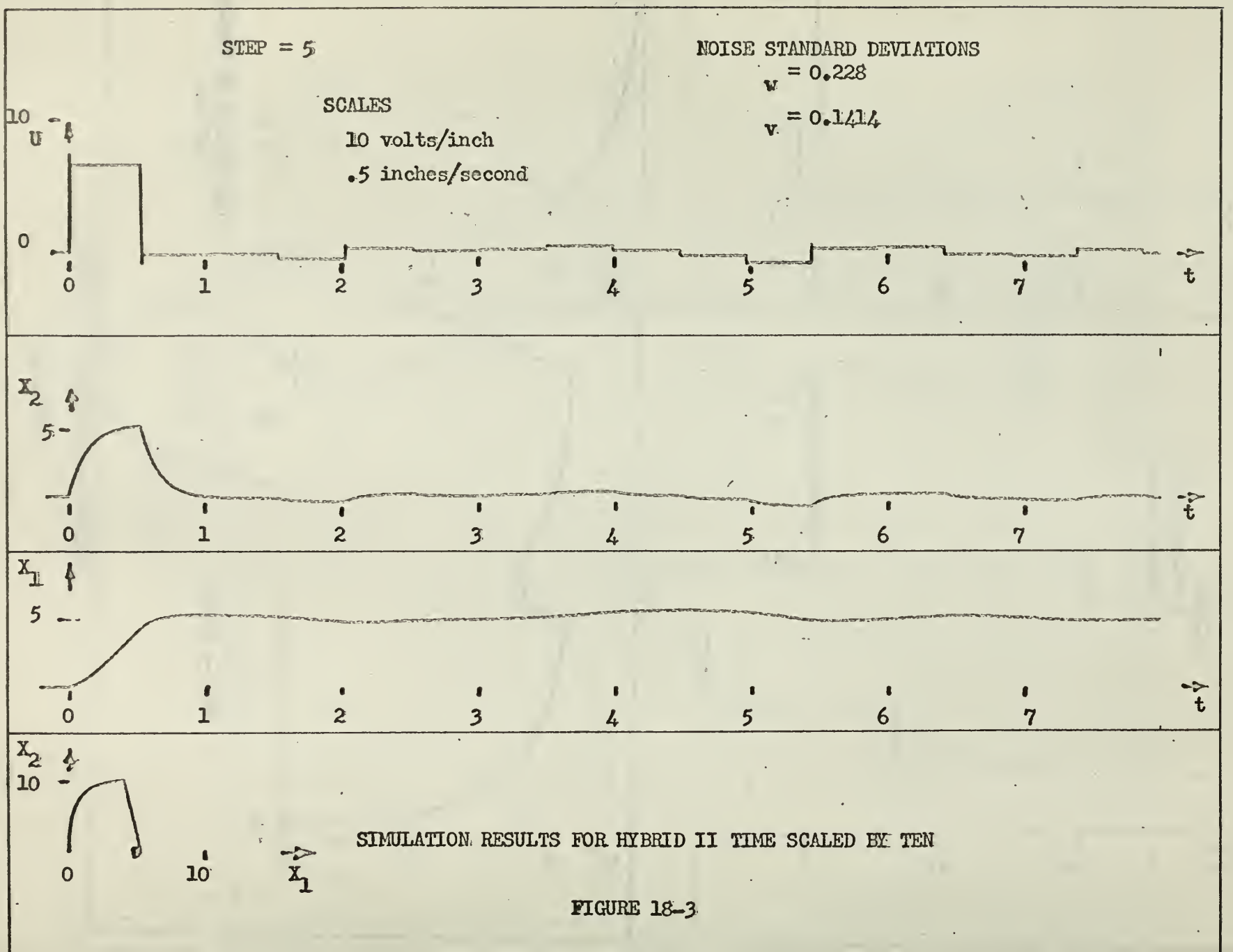
SIMULATION RESULTS FOR HYBRID II

FIGURE 18-1



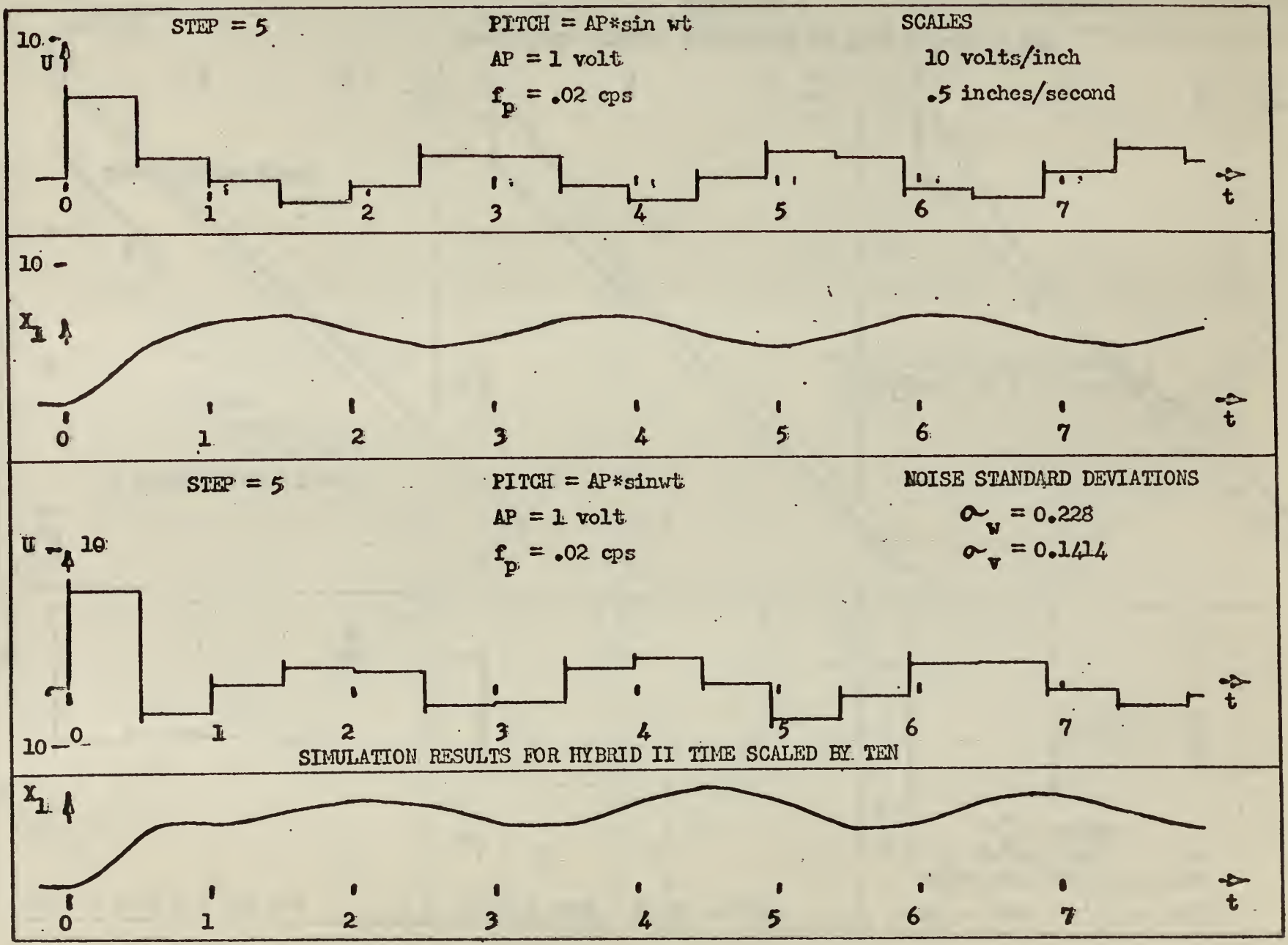
SIMULATION RESULTS FOR HYBRID II

FIGURE 18-2



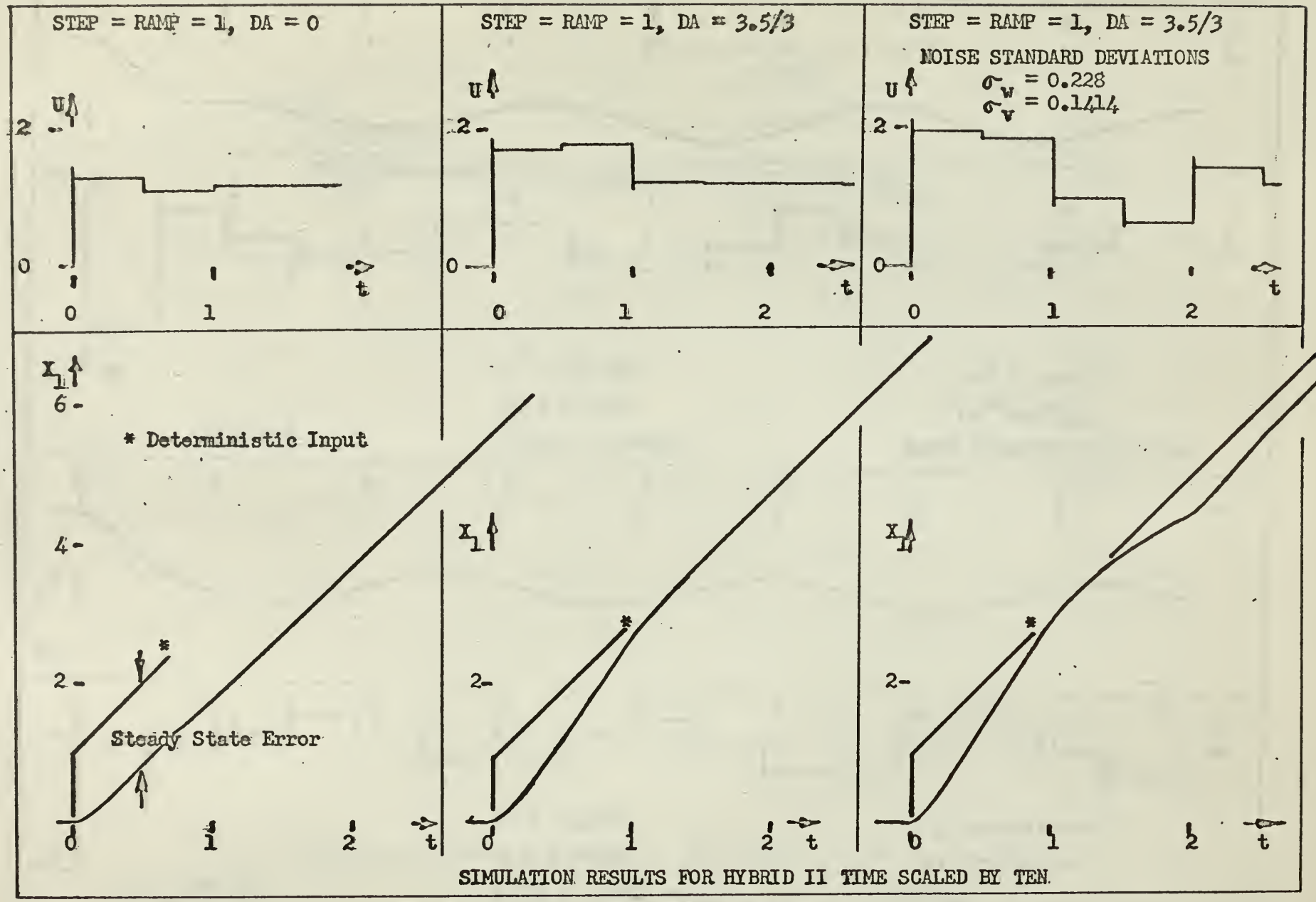
SIMULATION RESULTS FOR HYBRID II TIME SCALED BY TEN

FIGURE 18-3



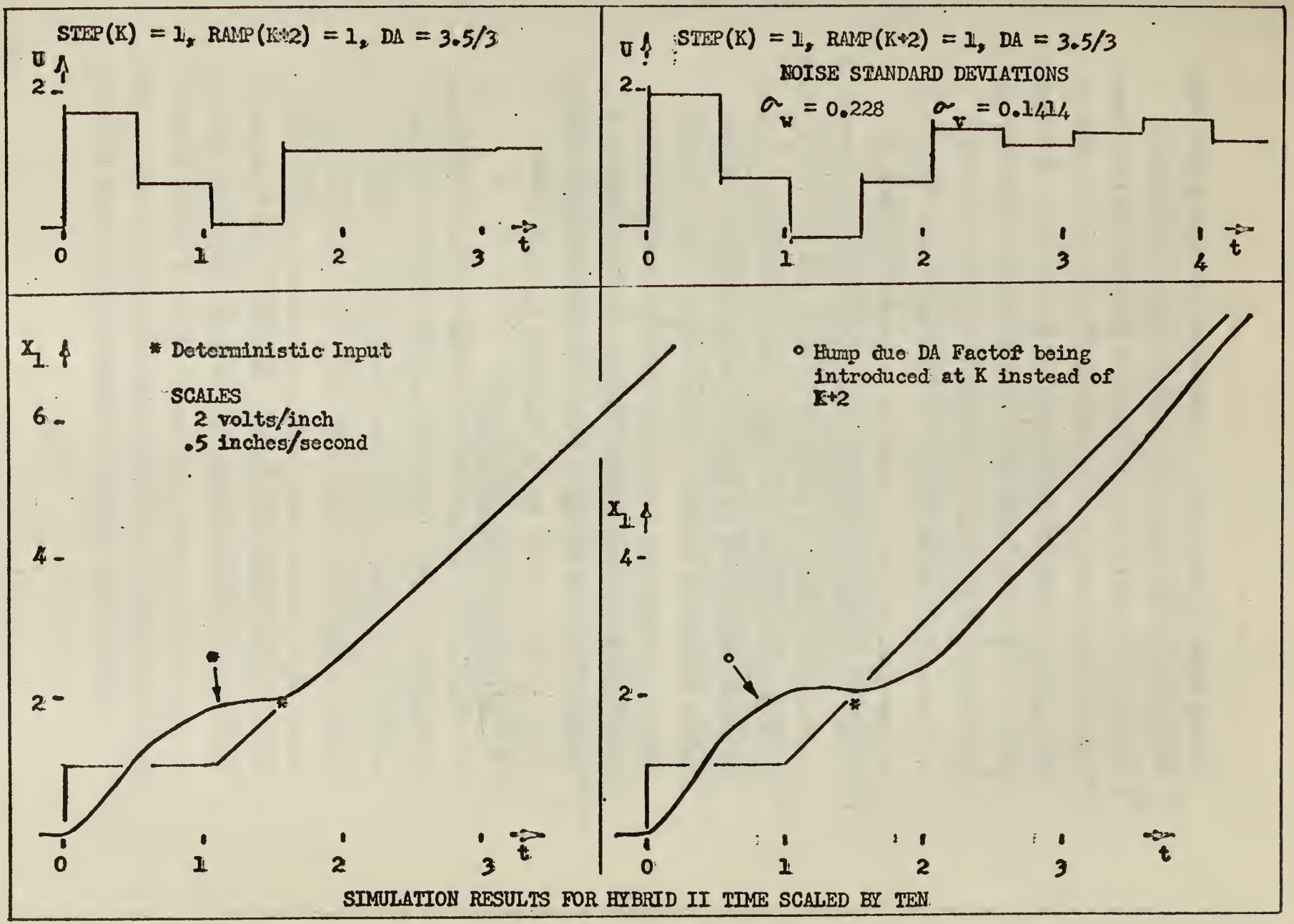
SIMULATION RESULTS FOR HYBRID II TIME SCALED BY TEN

FIGURE 18-4



SIMULATION RESULTS FOR HYBRID II TIME SCALED BY TEN.

FIGURE 18-5



SIMULATION RESULTS FOR HYBRID II TIME SCALED BY TEN

FIGURE 18-6

output of the control exists. In contrast, the time scaled step responses are the desired dead-beat, optimum responses. Figure 18-3 shows how the addition of environmental and measurement noise affect the gun response. In Figure 18-4 a pitch stabilization signal is introduced with a step input. The approximation of the pitch stabilization signal by the output of a zero order hold is seen to be satisfactory. In Figure 18-5 the requirement for the DA factor to ensure zero steady state error of this type one plant in response to a ramp is illustrated. Figure 18-6 depicts a typical situation of the weapon slewing in response to a step to a particular bearing, and tracking a target in response to a ramp. The hump is produced in this position response because the DA factor is introduced three samples before the ramp is initiated. In hind sight, a more judicious method of introducing this DA factor would be to detect the magnitude of the deterministic velocity input and calculate equation 7-4 each sample. Then, if the deterministic velocity vector is zero, the DA factor would also be zero.

19. Advantages of Digital Control

The use of a digital filter-controller has many significant advantages over the conventional analog system. In cases where more accuracy and higher sensitivity are required, digital compensation techniques excel over analog methods. In addition, virtually noise-free transmission of data in the form of digital numerical codes is possible. Non-linear programming and self-optimizing programming techniques can be utilized with the digital computer to provide a degree of flexibility not heretofor achievable with continuous control systems.

High operating speeds and efficient input/output procedures

have made the utilization of general purpose digital computers in real-time sampled data systems feasible. Time sharing of many equipments with the same computer is another important aspect. The use of vector-matrix notation for systems that can be represented by linear differential equations is ideally suited for digital computation, and has led to development of programs for analysis and control of systems.

20. Conclusions

The following statements are cited as being important conclusions to this hybrid simulation control problem:

A. The results of this hybrid simulation have demonstrated the feasibility and simplification in design which accompany the use of a general purpose computer as an integral component of the weapon control system.

B. The results also demonstrate that the application of a discrete filter-controller to provide control of a tactical weapon is feasible.

C. The results demonstrated that there was no difference between using constant or adaptive filter gains for this time-invariant weapon plant. This means that the simpler computational program of Hybrid I can be used, and thereby minimize the computation time required to effectively control the gun.

D. The application of good programming techniques to minimize hybridization limitations is important in providing optimum digital control.

E. The time-shared control of a number of weapons systems serially with the performance of other functions is feasible

for a general purpose, high speed computer having fast input/output capabilities and fast arithmetic subroutines.

BIBLIOGRAPHY

1. Jardine, F. D. Optimal Filter Design for Sampled Data Systems with Illustrative Examples, U. S. Naval Postgraduate School Master's Thesis, 1965.

2. Ogden, D. J. Optimum Digital Control Synthesis, U. S. Naval Postgraduate School Master's Thesis, 1965.

3. Schmidt, S. F. The Application of State Space Methods to Navigation Problems, Philco Technical Report No. 4, July 1964.

4. Demetry, J. S., Strum, R. D., Titus, H. A. A Case Study of the Discrete Optimum Filter-Controller Problem, U. S. Naval Postgraduate School Research Paper No. 57, November 1965.

5. Slaughter, J. B., Lackowski, D. H. Digital Control of a Tactical Weapon with a Shipboard G-P Computer, Navy Electronics Laboratory, 7th National Convention of Military Electronics, Proceedings, 1963.

APPENDIX I

STATE SPACE REPRESENTATION OF A SYSTEM

I-1 General Theory

The state of a system is defined as a set of variables which in conjunction with the system inputs completely describe the behaviour of the system. Consider a linear, time invariant system described by the second order differential equation:

$$\ddot{y} + a\dot{y} + c = f(t) \qquad \text{(I-1)}$$

The state variables x_1 and x_2 and the plant control are defined as follows:

$$\begin{aligned} x_1 &= y \\ x_2 &= \dot{x}_1 = \dot{y} \\ u &= f(t) - c \end{aligned} \qquad \text{(I-2)}$$

If $f(t)$ is known for all t greater than or equal to t_o and if the initial states are known, then the states can be determined for all time t. The second order linear differential equation is reduced to a set of first order state equations as follows:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -b*x_1 - a*x_2 + u \end{aligned}$$

or written in matrix notation as:

$$\dot{\underline{x}} = F\underline{x} + D\underline{u} \qquad \text{(I-3)}$$

where

$\underline{x}$ is an nxl vector of the system states

$\underline{u}$ is an rxl vector of the system inputs or controls

F is an nxn matrix of constants

D is an nxr matrix of constants

here and

$$F = \begin{bmatrix} 0 & 1 \\ b & -a \end{bmatrix} \qquad D = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

In general the output of the system may be a linear combination of the system states and controls, and a new set of output equations can be defined:

$$\underline{y} = A\,\underline{x} + B\,\underline{u}$$

where

y is a qxl vector of system outputs

A is a qxn matrix of constants

B is a qxr matrix of constants

It can be verified that the continuous solution to equation I-1 is given by

$$x(t) = \Phi(t-t_o)\,x(t) + \int_{t_o}^{t} \Phi(t-t_1)\,D\,u(t_1)\,dt_1 \qquad \text{(I-4)}$$

where

$\Phi(t-t_o)$ is called the state transition matrix and is defined as the inverse Laplace transform of $(sI-F)^{-1}$

and

$\int_{t_o}^{t} \Phi(t-t_o)\,D\,dt_1$ is called the state distribution matrix

for the case when the control u is the output of a zero order hold.

The discrete solution to equation I-4 is given by

$$\underline{x}(k+1) = \Phi\,\underline{x}(k) + \Delta\,u(k) \qquad \text{(I-5)}$$

where

$$\Phi = \epsilon^{FT}$$

$$\Delta = \int_{o}^{T} \epsilon^{F(t-t_1)}\,D\,dt_1$$

$T = t-t_o$, T is the sampling interval

and

$t_o = kT, \quad t = (k+1)T$

The advantages of describing an nth order linear differential

equation by a set of first order differential equations are:

(i) ease of solution of these first order equations by digital computer programming techniques

(ii) ease of correlation between state variables, flow graphs and analog computer simulation allowing unified study of sampled-data systems

(iii) ease of specification of initial conditions.

I-2. Flow Graphs

The flow graph of a linear, time invariant system can be obtained from the state equation I-3 or vice versa. Figure I-1 was drawn with the following ground rules:

(i) The system states are selected as outputs of integrators and the nodes selected as states have only one branch entering the node.

(ii) x_j is connected to $\dot{x}_{j-1}$ by a unity transmission branch

(iii) all feedback paths are from the integrator outputs of x_1, x_2 etc; to the node representing $\dot{x}_j$.

SAMPLE FLOW GRAPH

FOR

SECOND ORDER PLANT

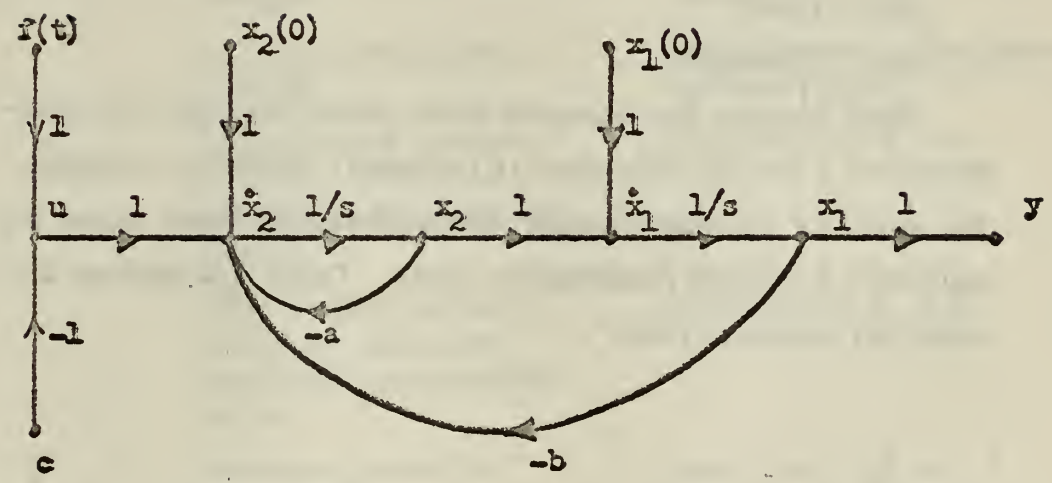


FIGURE I-1

APPENDIX II

FORTRAN 60 PROGRAMS

II-1. General Description

This Appendix contains the following Fortran 60 programs that were used to provide constants required for the hybrid simulation and also provide a means of verifying the results from the hybrid simulation:

(i) Optcon

(ii) Filter

(iii) Hybrid II

Each program has comment cards which explain their operation and a sample data deck is included. Table II-1 defines the terms for program 'Optcon' and outlines the three cases for selecting a desired performance index. Table II-2 defines the terms for program 'Filter'.

DEFINITION OF TERMS FOR PROGRAM OPTCON

Variable	Definition	Remarks
Nstage	number of stages	
N	order of system	
R	control effort weighting constant in the cost function	selected by the operator for desired performance
Q	state weighting matrix in the cost function	selected by the operator for desired performance
Δ	nxl control distribution vector	constant for a given time-invariant plant
Φ	nxn state transition matrix	constant for a given time-invariant plant
Ψ	intermediate variable matrix in recursive solution for controller gains	initial value $\Psi_o = 0$
P	intermediate variable matrix in recursive solutions for controller gains	initial value $P_o = Q$
$\underline{A}^T$	feedback controller gain vector	initial value $A_o^T = 0$
Case I	Minimum time or minimization of terminal states	$Q_o = I$, $Q^* = 0$, $R = 0$ in program 'Optcon'

For second order plant the minimum time performance index is

$$J(N) = x_1^2(N) + x_2^2(N)$$

Case II	Minimum effort or fuel	$Q_o = I$, $\hat{Q} = 0$, $R = 1$ in program 'Optcon'

For second order plant the minimum control effort performance index is

$$J(N) = x_1^2(n) + x_2^2(N) + \min_{u(k)} \sum_{k=1}^{N} u^2(k-1)$$

TABLE II-1

Case III	Minimum time and effort	$Q_o = I$, $Q^* \neq 0$, $R = 1$ in program 'Optcon'

For second order plant the minimum time and control effort performance index is

$$J(N) = \min_{u(k)} \sum_{k=1}^{N} \underline{x}^T(k)\underline{x}(k) + u^2(k-1)$$

Note: Q and R may be selected to satisfy different cost functions. Another method for selecting Q and R is given in reference [4].

TABLE II-1 (cont'd)

```
      PROGRAM OPTCON
      DIMENSION PHI(12,12),PSI(12,12),P(12,12),DEL(12),AT(12),
     1GM(12,12),Q(12,12),X(900),ITITLE(12),FM(16),EM(12),HM(12,12),
     1 Y1(900),Y2(900),Y3(900)

C    THIS PROGRAM UTILIZES A COST FUNCTION, J(N)=MINIMUM(SUM X(N)T*Q*X(N)+
C    SUM R*U(N-1)**2).  AN UNLIMITED NUMBER OF ITERATIONS MAY BE MADE AT
C    A COMPUTATION RATE OF 2000 PER MINUTE AFTER THE PROGRAM HAS BEEN
C    COMPILED.  THE OUTPUT OF THIS PROGRAM IS THE FEEDBACK GAIN MATRIX,
C    A TRANSPOSE.  THE FOLLOWING RECURSIVE EQUATIONS WERE DERIVED USING
C    DYNAMIC PROGRAMMING,
C    AT(K)=-(DELT*P(K-1)*PHI)/(DELT*P(K-1)*DEL+R)                    (1)
C    PSI(K)=PHI+DEL*AT(K)                                            (2)
C    P(K)=PSIT(K)*P(K-1)*PSI(K)+Q+R*A(K)*AT(K)                       (3)
C       P(0)=0,AT(0)=0, PSI(0)=0
C    EQUATIONS 1, 2, AND 3 CONSTITUTE THIS PROGRAM

C       CALL IN DATA AND INITIALIZE
        DO 1111 I=1,3
        READ 30,KASE
   30   FORMAT(I1)
        READ 1 ,N,NSTAGE
    1   FORMAT (8I10)
        READ 2,R,DT
    2   FORMAT((4F20.0))
        READ 2 ((Q(I,J),J=1,N),I=1,N)
C       PRINT THE DATA READ IN.
        PRINT 100
  100   FORMAT(1H1)
        PRINT 32,KASE
   32   FORMAT(9X,5HKASE= ,I3)
        PRINT 3,N,NSTAGE
    3   FORMAT(//9X,2HN=,I3,20X,7HNSTAGE=,I3)
        PRINT 4,R,DT
    4   FORMAT(/9X,2HR=,F15.11,2X,3HDT=,F15.11)
        PRINT 9,((Q(I,J),J=1,N),I=1,N)
    9   FORMAT(/9X,7HQ(I,J)=/(2F15.11))
        CALL PHIDEL(PHI,DEL,N,DT)
        DO 5 I=1,N
        DO 5 J=1,N
        GM(I,J)=0.0
        EM(I)=0.0
        FM(I)=0.0
    5   P(I,J)=Q(I,J)
C       INITIALIZE P(0) FOR CASE TWO
        IF(KASE-2) 35,36,35
   36   P(1,1)=1.0
        P(2,2)=1.0
        P(3,3)=1.0
```

PROGRAM OPTCON CONTINUED

```
   35 CONTINUE
      PRINT 19
   19 FORMAT(//2X,6HNSTAGE,4X,7HAT(1,1),3X,7HAT(1,2),4X,6HP(1,1),
     14X,6HP(1,2),4X,6HP(2,1),4X,6HP(2,2))
C     CALCULATE  AT(J)
      DO  22 KK=1,NSTAGE
      DEN=0.0
      DO6 I=1,N
      DO6 J=1,N
    6 EM(I)=EM(I)+DEL(J)*P(J,I)
      DO8 I=1,N
      DO7 J=1,N
    7 FM(I)=FM(I)+EM(J)*PHI(J,I)
    8 DEN=DEN+EM(I)*DEL(I)
      DEN=-DEN-R
      DO10I=1,N
      AT(I)=FM(I)/DEN
      FM(I)=0.0
   10 EM(I)=0.0
C     CALCULATE            PSI(K,I,J)
      DO13I=1,N
      DO13J=1,N
   13 PSI(I,J)=PHI(I,J)+DEL(I)*AT(J)
C     CALCULATE            P(K,I,J)
      DO 15 I=1,N
      DO 15 J=1,N
      DO 15 L=1,N
   15 GM(I,J)=GM(I,J)+PSI(L,I)*P(L,J)
      DO 17I=1,N
      DO 17J=1,N
      DO 16 L=1,N
   16 HM(I,J)=HM(I,J)+GM(I,L)*PSI(L,J)

C     CASE 1  TERMINAL CONTROL, OMIT Q(I,J) IN EQUATION FOR P(I,J)
C     CASE2  MINIMUM FUEL       OMIT Q(I,J) IN EQUATION FOR P(I,J)
      IF(KASE-2) 31,31,33
   31 P(I,J)=HM(I,J)+R*AT(I)*AT(J)
      GO TO 17

C     CASE THREE MINIMIZATION OF TIME AND FUEL
   33 P(I,J)=HM(I,J)+Q(I,J)+R*AT(I)*AT(J)
   17 HM(I,J)=0.0
      DO18 I=1,N
      DO18 J=1,N
   18 GM(I,J)=0.0
      PRINT 20,KK,AT(1),AT(2),P(1,1),P(1,2),P(2,1),P(2,2)
   20 FORMAT(I5,(6F10.6))
   22 CONTINUE
 1111 CONTINUE
      END
```

PROGRAM OPTCON CONTINUED

```
      SUBROUTINE PHIDEL(PHI,DEL,N,DT)
C     VALID ONLY FOR A CONSTANT F MATRIX
      DIMENSION F(12,12),PHI(12,12),TERM(12,12),WORM(12,12)
     1, DEL(12),DELM(12,12),TELM(12,12),DELP(12,12),D(12)
    1 FORMAT ((8F10.0))
      READ1,((F(IR,IC),IC=1,N),IR=1,N)
      READ 1 (D(I),I=1,N)
 1003 PRINT 399,DT,((F(IR,IC),IC=1,N),IR=1,N)
  399 FORMAT(///3HDT=,1F5.3/   ,7HF(I,J)=/,((2F8.2)))
      PRINT 3991 (D(I),I=1,N)
 3991 FORMAT(/   5HD(I)=/(2F8.2))
      NFINAL=1
      TM=0.0
      DO 400 IR=1,N
      DO 400 IC=1,N
      TERM(IR,IC)=0.0
      WORM(IR,IC)=0.0
      TERM(IR,IR)=1.0
      TELM(IR,IC)=TERM(IR,IC)*DT
      DELP(IR,IC)=TELM(IR,IC)
      DELM(IR,IC)=0.0
      DEL(IR)=0.0
  400 PHI(IR,IC)=TERM(IR,IC)
    4 TM=1.0+TM
      DO 500    IR=1,N
      DO 500    IC=1,N
      DO 500    JN=1,N
      DELM(IR,IC)=DELM(IR,IC)-TELM(IR,JN)*F(JN,IC)*DT/(TM+1.0)
  500 WORM(IR,IC)=TERM(IR,JN)*F(JN,IC)*DT/TM+WORM(IR,IC)
      DO 401 IR=1,N
      DO 401 IC=1,N
      TERM(IR,IC)=WORM(IR,IC)
      TELM(IR,IC)=DELM(IR,IC)
      DELP(IR,IC)=DELP(IR,IC)+TELM(IR,IC)
      PHI(IR,IC)=PHI(IR,IC)+TERM(IR,IC)
      DELM(IR,IC)=0.0
  401 WORM(IR,IC)=0.0
      ABC=0.0
      DO 2I=1,N
      DO 2J=1,N
      AA=TERM(I,J)
      AB=ABSF(AA)
      IF(ABC-AB)3,3,2
    3 ABC=AB
    2 CONTINUE
      IF(0.000000005-ABC)5,5,6
    5 GO TO 4
    6 PRINT 11,((PHI(I,J),J=1,N),I=1,N)
   11 FORMAT(/9X,9HPHI(I,J)=/(2F15.11))
      DO 600 I=1,N
```

PROGRAM OPTCON CONTINUED

```
      DO 600 K=1,N
      DO 600 J=1,N
  600 DEL(I)=DEL(I)+PHI(I,J)*DELP(J,K)*D(K)
      PRINT 12,(DEL(I),I=1,N)
   12 FORMAT(9X,7HDEL(I)=/(1F15.11))
      END
      END

1
          2          20
                       1.0
1.0                                                          1.0
           1.0                      -3.5
            3.0
2
          2          20
1.0                    1.0
1.0                                                          1.0
           1.0                      -3.5
            3.0
3
          2          20
1.0                    1.0
1.0                                                          1.0
           1.0                      -3.5
            3.0
```

DEFINITION OF TERMS FOR PROGRAM FILTER

Variable	Definition	Remarks
Φ	nxn state transition matrix	constant for a given time-invariant plant
Δ	nxl control distribution vector	constant for a given time-invariant plant
G	nxl optimum filter gain matrix	
H	plant observability matrix	order of matrix depends upon number of plant observables
R	measurement noise covariance matrix	value selected by operator
Q	random excitation distribution covariance matrix	value selected by operator
P	error covariance matrix	initial value selected by operator
$\underline{X}$	nxl vector of plant states	
$\underline{Y}$	nxl vector of plant non noisy observable states	
$\underline{Z}$	nxl vector of noisy plant observable states	
$\hat{\underline{X}}$	nxl vector of the predicted value of $\underline{X}(k)$, based on the previous observation $\underline{X}(k-1)$	
$\hat{\underline{Z}}$	nxl vector of the predicted value of $\underline{Z}(k)$, based on the previous observation $\underline{Z}(k)$	
$\overset{*}{\underline{X}}$	nxl vector of the best estimate of $\underline{X}(k)$, based on the current observation of $\underline{Z}(k)$	
W(k)	random disturbance of environment or excitation noise	assumed normal with mean zero

TABLE II-2

Variable	Definition	Remarks
V(k)	random disturbance of measurement noise	assumed normal with mean zero
σ_w	standard deviation of excitation noise	
σ_v	standard deviation of measurement noise	

TABLE II-2(cont'd)

```
      PROGRAM FILTER
C     D1 ORDER OF SYSTEM IN I2 FORMAT
C     D2 SAMPLING INTERVAL IN 8F10.0 FORMAT
C     D3 F MATRIX BY ROWS IN 8F10.0 FORMAT
C     D4 D MATRIX BY COLUMN IN 8F10.0 FORMAT
C     D5  VARIANCE OF EXCITATION NOISE SIGWSQ IN 8F10.6 FORMAT
C     D6 R COVARIANCE MATRIX OF MEASUREMENT NOISE IN 8F10.6 FORMAT
C      PHI SYSTEM TRANSITION MATRIX
C      DEL DISTRIBUTION MATRIX
C      G  OPTIMUM GAIN MATRIX
C      H  OBSERVABLE MATRIX
C      P BEST ESTIMATE OF ERROR COVARIANCE MATRIX
C      Q  EXCITATION NOISE COVARIANCE MATRIX
      DIMENSION P(12,12),Q(12,12),H(12,12),R(12,12),G(12,12),PHIT(12,12)
     1,PHI(12,12),DEL(12),DELDELT(12,12),DELS(12,12),DELST(12,12),
     2X(12),XHAT(12),YHAT(12) ,PNEW(12,12)
      READ 33, N
   33 FORMAT (I2)
      READ 2,DT
    2 FORMAT(8F10.0)
      PRINT 1003
 1003 FORMAT(1H1)
      CALL PHIDEL(PHI,DEL,N,DT)
      DO 1001 LL=1,4
      READ 20,SIGWSQ
   20 FORMAT(F10.6)
      DO 1001 LK=1,4
C     INIALIZE MATRICES FOR EACH RUN
      DO 10 I=1,N
      DO 10 J=1,N
      PNEW(I,J)=0.0
      P(I,J)=0.0
      G(I,J)=0.0
      Q(I,J)=0.0
       DELDELT(I,J)=0.0
      DELS(I,J)=0.0
   10 DELST(I,J)=0.0
      READ 2001,R(1,1)
 2001 FORMAT(8F10.6)
      DO 30 I=1,N
   30 DELS(I,1)=DEL(I)
      CALL TRANS(DELS,N,N,DELST)
      CALL PROD(DELS,DELST,N,N,N,DELDELT)
      DO 40 I=1,N
      DO 40 J=1,N
   40 Q(I,J)=SIGWSQ*DELDELT(I,J)
      SIGW=SQRTF(SIGWSQ)
      PRINT 1004,SIGW
 1004 FORMAT(/ 10X,5HSIGW= /,E20.8)
```

PROGRAM FILTER CONTINUED

```
      P(1,1)=.02
      P(1,2)=0.0
      P(2,1)=0.0
      P(2,2)=1.0
      PRINT 2002,R(1,1),((Q(I,J),J=1,N),I=1,N),((P(I,J),J=1,N),I=1,N)
 2002 FORMAT(///20X,8HR(1,1)= ,F10.6/20X,8HQ(I,J)= ,4F10.6/20X,8HP(I,J)
     1 ,4F10.6)
      H(1,1)=1.
      H(1,2)=0.0
      PRINT 700
  700 FORMAT(//5X,3HG11,7X,3HG12,7X,3HG21,7X,3HG22,7X,3HP11,7X,3HP12,7X
     1 3HP21,7X,3HP22/)
      DO 1000 KK=1,40
      CALL GP(H,PHI,P,Q,R,2,1,G,PNEW)
      PRINT 800,((G(I,J),J=1,N),I=1,N),((PNEW(I,J),J=1,N),I=1,N)
  800 FORMAT (10F10.5)
      DO 11 I=1,N
      DO 11 J=1,N
   11 P(I,J)=PNEW(I,J)
 1000 CONTINUE
      PRINT 1005
 1005 FORMAT(1H1)
 1001 CONTINUE
      END


      SUBROUTINE GP(H,PHI,P,Q,R,KN,KP,G,PNEW)
      DIMENSION H(12,12),PHI(12,12),P(12,12),Q(12,12),R(12,12),G(12,12)
     1PNEW(12,12),HT(12,12),TV1(12,12),TV2(12,12)
      CALL TRANS(H,KP,KN,HT)
      CALL PROD(P,HT,KN,KN,KP,TV1)
      CALL PROD(H,TV1,KP,KN,KP,TV2)
      CALL ADD(TV2,R,KP,KP,TV1)
      CALL RECIP(KP,.000000000001,TV1,TV2,KER)
      IF(KER-2) 101,110,101
  110 PRINT 111
  111 FORMAT(5HKER=2)
  101 CALL PROD(HT,TV2,KN,KP,KP,TV1)
      CALL PROD(P,TV1,KN,KN,KP,G)
      CALL PROD(H,P,KP,KN,KN,TV1)
      CALL PROD(G,TV1,KN,KP,KN,TV2)
      DO 102 I=1,KN
      DO 102 J=1,KN
  102 TV2(I,J)=-TV2(I,J)
      CALL ADD(P,TV2,KN,KN,TV1)
      CALL PROD(PHI,TV1,KN,KN,KN,TV2)
      CALL TRANS(PHI,KN,KN,TV1)
      CALL PROD(TV2,TV1,KN,KN,KN,PNEW)
      CALL ADD(PNEW,Q,KN,KN,PNEW)
      END
```

PROGRAM FILTER CONTINUED

```
      SUBROUTINE TRANS(A,N,M,C)
      DIMENSION A(12,12),C(12,12)
      DO 153 I = 1,N
      DO 153 J=1,M
  153 C(J,I) = A(I,J)
      END


      SUBROUTINE RECIP(N,EP,A,X,KER)
      DIMENSION A(12,12),X(12,12)
      DO 1 I=1,N
      DO 1 J=1,N
    1 X(I,J)=0.0
      DO 2 K=1,N
    2 X(K,K)=1.0
   10 DO 34 L=1,N
      KP=0
      Z=0.0
      DO 12 K=L,N
      IF(Z-ABSF(A(K,L)))11,12,12
   11 Z=ABSF(A(K,L))
      KP = K
   12 CONTINUE
      IF(L-KP)13,20,20
   13 DO 14 J=L,N
      Z=A(L,J)
      A(L,J)=A(KP,J)
   14 A(KP,J)=Z
      DO 15 J=1,N
      Z=X(L,J)
      X(L,J)=X(KP,J)
   15 X(KP,J)=Z
   20 IF(ABSF(A(L,L))-EP)50,50,30
   30 IF(L-N)31,34,34
   31 LP1=L+1
      DO 36 K=LP1,N
      IF(A(K,L))32,36,32
   32 RATIO=A(K,L)/A(L,L)
      DO 33 J=LP1,N
   33 A(K,J)=A(K,J)-RATIO*A(L,J)
      DO 35 J=1,N
   35 X(K,J)=X(K,J)-RATIO*X(L,J)
   36 CONTINUE
   34 CONTINUE
   40 DO 43 I=1,N
      II=N+1-I
      DO 43 J=1,N
      S=0.0
```

PROGRAM FILTER CONTINUED

```
      IF(II-N)41,43,43
   41 IIP1=II+1
      DO 42 K=IIP1,N
   42 S=S+A(II,K)*X(K,J)
   43 X(II,J)=(X(II,J)-S)/A(II,II)
      KER=1
      RETURN
   50 KER=2
      END


      SUBROUTINE PHIDEL(PHI,DEL,N,DT)
C     VALID ONLY FOR A CONSTANT F MATRIX
      DIMENSION F(12,12),PHI(12,12),TERM(12,12),WORM(12,12)
     1, DEL(12),DELM(12,12),TELM(12,12),DELP(12,12),D(12)
      READ1,((F(IR,IC),IC=1,N),IR=1,N)
    1 FORMAT ((8F10.0))
      READ 1 (D(I),I=1,N)
 1003 PRINT 399,DT,((F(IR,IC),IC=1,N),IR=1,N)
  399 FORMAT(///3HDT=,1F5.3///,7HF(I,J)=/,((8F8.2)))
      PRINT 3991 (D(I),I=1,N)
 3991 FORMAT(////5HD(I)=/(8F8.2))
      NFINAL=1
      TM=0.0
      DO 400 IR=1,N
      DO 400 IC=1,N
      TERM(IR,IC)=0.0
      WORM(IR,IC)=0.0
      TERM(IR,IR)=1.0
      TELM(IR,IC)=TERM(IR,IC)*DT
      DELP(IR,IC)=TELM(IR,IC)
      DELM(IR,IC)=0.0
      DEL(IR)=0.0
  400 PHI(IR,IC)=TERM(IR,IC)
    4 TM=1.0+TM
      DO 500   IR=1,N
      DO 500   IC=1,N
      DO 500   JN=1,N
      DELM(IR,IC)=DELM(IR,IC)-TELM(IR,JN)*F(JN,IC)*DT/(TM+1.0)
  500 WORM(IR,IC)=TERM(IR,JN)*F(JN,IC)*DT/TM+WORM(IR,IC)
      DO 401 IR=1,N
      DO 401 IC=1,N
      TERM(IR,IC)=WORM(IR,IC)
      TELM(IR,IC)=DELM(IR,IC)
      DELP(IR,IC)=DELP(IR,IC)+TELM(IR,IC)
      PHI(IR,IC)=PHI(IR,IC)+TERM(IR,IC)
      DELM(IR,IC)=0.0
  401 WORM(IR,IC)=0.0
      ABC=0.0
      DO 2I=1,N
```

PROGRAM FILTER CONTINUED

```
      DO 2J=1,N
      AA=TERM(I,J)
      AB=ABSF(AA)
      IF(ABC-AB)3,3,2
    3 ABC=AB
    2 CONTINUE
      IF(0.000000005-ABC)5,5,6
    5 GO TO 4
    6 PRINT 502,((PHI(IR,IC),IC=1,N),IR=1,N)
  502 FORMAT(////9X,8HPHI(I,J)///(6E20.8))
      DO 600 I=1,N
      DO 600 K=1,N
      DO 600 J=1,N
  600 DEL(I)=DEL(I)+PHI(I,J)*DELP(J,K)*D(K)
      PRINT 503 (DEL(I),I=1,N)
  503 FORMAT(////9X,6HDEL(I)//(6E20.8)//)
      END


      SUBROUTINE ADD (A,B,N,M,C)
      DIMENSION A(12,12),B(12,12),C(12,12)
      DO 152 I=1,N
      DO 152 J=1,M
  152 C(I,J) = A(I,J) + B(I,J)
      END


      SUBROUTINE PROD (A,B,N,M,L,C)
      DIMENSION A(12,12),B(12,12),C(12,12)
      DO 151 I=1,N
      DO 151 J=1,L
      C(I,J) =0.
      DO 151 K = 1,M
  151 C(I,J) = C(I,J) + A(I,K)*B(K,J)
      END
      END
```

```
 2
1.0
          1.0                    -3.5
          3.0
.052
 .0
.02
 .1
1.0
```

```
      PROGRAM HYBRID II
C
C     THIS PROGRAM USES VARIABLE FILTER GAINS
C     OTHERWISE IT IS ESSENTIALLY PROGRAM HYBRID 1
C
C     THIS PROGRAM CLOSES THE LOOP ON THE OPTIMUM FILTER-CONTROLLER
C     PROBLEM. IT ASSUMES THAT STABLE CONTROLLER AND FILTER GAIN MATRICE
C     HAVE BEEN COMPUTED ON THE BASIS OF DESIRED RESPONSE AND THE
C     STATISTICAL PROPERTIES OF THE ANTICIPATED RANDOM DISTURBANCE AND
C     MEASUREMENT NOISE. THE SYSTEM MAY BE DRIVEN EITHER BY INITIAL
C     CONDITIONS OR BY STEP OR RAMP INPUTS,OR BY ANY COMBINATION OF
C     THESE.
C     THE PROGRAM SOLVES THE FOLLOWING EQUATIONS
C         Y(K)=H*X(K)
C         Z(K)=Y(K)+V(K)
C     XS(K)=(I-GH)PHI*XS(K-1)+(I-GH)*DEL*AT(XS(K-1)-DINP(K-1))+G*Z(K)
C     X(K+1)=PHI*X(K)+DEL*W(K)+DEL*AT*(XS(K)-DINP(K))
C     WHERE V IS MEASUREMENT NOISE,W IS THE RANDOM DISTURBANCE,AND
C     DINP IS THE DETERMINISTIC INPUT
      DIMENSION X(12,12),XS(12,12),SIGV(12),Y(12,12),Z(12,12),PHI(12,12)
     1,DEL(12,12),H(12,12),AT(12,12),G(12,12),TEMP1(12,12),TEMP2(12,12),
     2TEMP3(12,12),TEMP4(12,12),TELP(12,12),TELP1(12,12),TELP2(12,12),
     3V(12,12),DINP(12,12),IT(12),TMF(900),X1(900),XS1(900),X2(900),
     4XS2(900),U(12,12),B(12,12),P(12,12) ,R(12,12),B11(900),UU(900)
      READ 1,(IT(I),I=1,6)
      READ1,(IT(I),I=7,12)
    1 FORMAT(6A8)
      DO 1111 K=1,5
C     INPUT SOME CONSTANTS, AS NOTED
      T=1.0
C     KN IS SYSTEM ORDER. KP IS THE NUMBER OF OBSERVABLES.
      KN=2
      KP=1
      PIE=3.1415926
      AR=0.5
      THETAR=PIE/5.0
      AP=0.1
      THETAP=PIE/2.5
C     STEP IS MAGNITUDE OF STEP,RAMP IS MAGNITUDE OF RAMP, ROLL
C      AND PITCH SIGNALS ARE OUTPUT OF ZERO ORDER HOLD, CONSTANT OVER SAM
      STEP=4.0
      RAMP=0.0
C     DT IS INVOLVED IN UPDATING ROLL OR PITCH INPUTS
      DT=0.0
C     DR IS INVOLVED IN UPDATING RAMP INPUT
      DR=0.0
C     DA IS USED TO REDUCE STEADY STATE ERROR TO RAMP INPUT TO ZERO
C     FOR THIS TYPE ONE SYSTEM
      DA=0.0
```

PROGRAM HYBRID II CONTINUED

```
C       SETTING DINP=0.0 AT THIS POINT INSURES THAT NO DETERMINISTIC
C       INPUTS EXIST PRIOR TO TIME = ZERO
        DINP(1,1)=0.0
        DINP(2,1)=0.0
C       INITIALIZE MATRICES
        DO 66 I=1,KN
        DO 66 J=1,KN
        Y(I,J)=0.0
        Z(I,J)=0.0
        U(I,J)=0.0
        B(I,J)=0.0
        AT(I,J)=0.0
        G(I,J)=0.0
        H(I,J)=0.0
        DEL(I,J)=0.0
   66   PHI(I,J)=0.0
        DO 65 I=1,KN
   65   SIGV(I)=0.0
        READ 2,SIGV(1),SIGW
    2   FORMAT(2F10.6)
        R(1,1)=SIGV(1)*SIGV(1)
        SIGWSQ=SIGW*SIGW
C       CARDS ABOVE PROVIDE FOR SETTING DESIRED VARIANCES IN W AND V.
C       SIGW IS THE STANDARD DEVIATION OF W. SIMILARLY FOR SIGV.
C
C       INPUT MATRICES
        PHI(1,1)=1.0
        PHI(1,2)=0.277086
        PHI(2,1)=0.0
        PHI(2,2)=0.030197
        DEL(1,1)=0.619640
        DEL(2,1)=0.831259
        H(1,1)=1.0
        H(1,2)=0.0
        P(1,1)=0.02
        P(1,2)=0.0
        P(2,1)=0.0
        P(2,2)=1.0
        AT(1,1)=-1.2030
        AT(1,2)=-0.3426
C
C       THE FOLLOWING CARD INITIALIZES THE RANDOM NUMBER GENERATOR
        NUNIF=1220703125
        PRINT 3
    3   FORMAT(1H1)
        PRINT 800
  800   FORMAT(//4X,4HX(1),4X,5HXS(1),3X,4HX(2),4X,5HXS(2),3X,1HV,7X,1HW,
       17X,3HG11,5X,3HG12,5X,4HAT11,4X,4HAT12,4X,1HU,7X,3HB11,5X,3HB21,2X,-
       26HDINP11,2X,6HDINP21 //)
```

PROGRAM HYBRID II CONTINUED

```
C       PLANT INITIAL CONDITIONS
        X(2,1)=0.0
        X(1,1)=0.0
C       FILTER INITIAL CONDITIONS
        CALL PROD(H,X,KP,KN,KP,Y)
        DO 12 I=1,KP
        CALL RNDEV(NUNIF,DEV)
   12   V(I,1)=SIGV(I)*DEV
        CALL ADD(Y,V,KP,1,Z)
        XS(1,1)=Z(1,1)
        XS(2,1)=0.0

        LL=70
        DO 900 KK=1,LL
        CALL FILTER(KN,KP,PHI,DEL,SIGWSQ,R,H,P,G)
        CALL PROD(H,X,KP,KN,1,Y)
        DO 10 I=1,KP
        CALL RNDEV(NUNIF,DEV)
   10   V(I,1)=SIGV(I)*DEV
        CALL ADD(Y,V,KP,1,Z)
        CALL PROD(PHI,XS,KN,KN,1,TEMP1)
        CALL PROD(G,H,KN,KP,KN,TEMP2)
        DO 11 I=1,KN
        DO 11 J=1,KN
   11   TEMP2(I,J)=-TEMP2(I,J)
        CALL PROD(TEMP2,TEMP1,KN,KN,1,TEMP3)
        CALL ADD(TEMP1,TEMP3,KN,1,TEMP3)
        CALL ADD(XS,DINP,KN,1,TELP)
        CALL PROD(AT,TELP,1,KN,1,TEMP1)
        TEMP1(1,1)=TEMP1(1,1)+DA
        CALL PROD(DEL,TEMP1,KN,1,1,TELP)
        CALL PROD(TEMP2,TELP,KN,KN,1,TELP1)
        CALL ADD(TELP,TELP1,KN,1,TELP1)
        CALL PROD(G,Z,KN,KP,1,TELP2)
        CALL ADD(TEMP3,TELP1,KN,1,XS)
        CALL ADD(XS,TELP2,KN,1,XS)

C       TIME ZERO IS FOR KK-1
        IF(KK-4) 17,17,18
   18   CONTINUE
        RAMP=1.0
        DR=DR+1.0
        DA=RAMP*(3.5/3.0)
   17   CONTINUE
        DINP(1,1)=-(STEP+RAMP*DR+AR*SINF(THETAR*DT))
        DINP(2,1)=-RAMP
        TME(KK)=KK-1
        X1(KK)=X(1,1)
        XS1(KK)=XS(1,1)
```

PROGRAM HYBRID II CONTINUED

```
      X2(KK)=X(2,1)
      XS2(KK)=XS(2,1)
      B11(KK)=B(1,1)
      UU(KK)=U(1,1)
      PRINT 801,X(1,1),XS(1,1),X(2,1),XS(2,1),V(1,1),W,G(1,1),G(2,1),
     1AT(1,1),AT(1,2),U(1,1),B(1,1),B(2,1),DINP(1,1),DINP(2,1)
  801 FORMAT(15F8.4)
      CALL RNDEV(NUNIF,DEV)
      W=SIGW*DEV
      CALL PROD(PHI,X,KN,KN,1,TEMP1)
      DO 803 I=1,KN
  803 TEMP2(I,1)=W*DEL(I,1)
      CALL ADD(XS,DINP,KN,1,TELP)
      DO 13 I=1,KN
      DO 13 J=1,KN
   13 B(I,J)=TELP(I,J)
      CALL PROD(AT,TELP,1,KN,1,TELP1)
      DO 14 I=1,KN
      DO 14 J=1,KN
   14 U(I,J)=TELP1(I,J)
      TELP1(1,1)=TELP1(1,1)+DA
      CALL PROD(DEL,TELP1,KN,1,1,TELP2)
      CALL ADD(TEMP1,TEMP2,KN,1,X)
      CALL ADD(X,TELP2,KN,1,X)
  900 CONTINUE
       MC=1
      LA=4HX1
      CALL DRAW(LL,TME,X1 ,MC,0,LA,IT,10.0,10.,3,0,2,2,0,10,1,LAST)
      MC=2
      LA=4HXS1
      CALL DRAW(LL,TME,XS1,MC,0,LA,IT,0.0,0.0,3,0,2,2,0,10,1,LAST)
      MC=2
      LA=4HX2
      CALL DRAW(LL,TME,X2 ,MC,0,LA,IT,0.0,0.0,3,0,2,2,0,10,1,LAST)
      MC=2
      LA=4HXS2
      CALL DRAW(LL,TME,XS2,MC,0,LA,IT,0.0,0.0,3,0,2,2,0,10,1,LAST)
      MC=3
      LA=4HB11
      CALL DRAW(LL,TME,B11,MC,0,LA,IT,0.0,0.0,3,0,2,2,0,10,1,LAST)
      MC=1
      LA=4HX1-T
      CALL DRAW(LL,TME,X1 ,MC,0,LA,IT,10.0,10.,3,0,2,2,0,10,1,LAST)
      MC=2
      LA=4HU-T
      CALL DRAW(LL,TME,UU ,MC,0,LA,IT,1.0,1.0,3,0,2,2,0,10,1,LAST)
      MC=2
      LA=4HX2-T
      CALL DRAW(LL,TME,X2 ,MC,0,LA,IT,1.0,1.0,3,0,2,2,0,10,1,LAST)
      MC=3
```

```
      LA=4HX1X2
      CALL DRAW(LL,X1 ,X2 ,MC,0,LA,IT,1.0,1.0,3,0,2,2,0,10,1,LAST)
 1111 CONTINUE
      END

      SUBROUTINE PROD (A,B,N,M,L,C)
      DIMENSION A(12,12),B(12,12),C(12,12)
      DO 151 I=1,N
      DO 151 J=1,L
      C(I,J)=0.
      DO 151 K=1,M
  151 C(I,J)=C(I,J)+A(I,K)*B(K,J)
      END

      SUBROUTINE ADD (A,B,N,M,C)
      DIMENSION A(12,12),B(12,12),C(12,12)
      DO 152 I=1,N
      DO 152 J=1,M
  152 C(I,J)=A(I,J)+B(I,J)
      END


      SUBROUTINE FILTER(N,KP,PHI,DEL,SIGWSQ,R,H,P,G)
C      PHI SYSTEM TRANSITION MATRIX
C      DEL DISTRIBUTION MATRIX
C      G  OPTIMUM GAIN MATRIX
C      H  OBSERVABLE MATRIX
C      P BEST ESTIMATE OF ERROR COVARIANCE MATRIX
C      Q  EXCITATION NOISE COVARIANCE MATRIX
      DIMENSION P(12,12),Q(12,12),H(12,12),R(12,12),G(12,12),PHIT(12,12)
     1,PHI(12,12),DEL(12),DELDELT(12,12),DELS(12,12),DELST(12,12),
     2PNEW(12,12)
      DO 30 I=1,N
   30 DELS(I,1)=DEL(I)
      CALL TRANS(DELS,N,N,DELST)
      CALL PROD(DELS,DELST,N,N,N,DELDELT)
      DO 40 I=1,N
      DO 40 J=1,N
   40 Q(I,J)=SIGWSQ*DELDELT(I,J)
      CALL GP(H,PHI,P,Q,R,N,KP,G,PNEW)
      DO 11 I=1,N
      DO 11 J=1,N
   11 P(I,J)=PNEW(I,J)
      END


      SUBROUTINE GP(H,PHI,P,Q,R,KN,KP,G,PNEW)
      DIMENSION H(12,12),PHI(12,12),P(12,12),Q(12,12),R(12,12),G(12,12),
     1PNEW(12,12),HT(12,12),TV1(12,12),TV2(12,12)
      CALL TRANS(H,KP,KN,HT)
```

PROGRAM HYBRID II CONTINUED

```
      CALL PROD(P,HT,KN,KN,KP,TV1)
      CALL PROD(H,TV1,KP,KN,KP,TV2)
      CALL ADD(TV2,R,KP,KP,TV1)
      CALL RECIP(KP,.0000000000001,TV1,TV2,KER)
      IF(KER-2) 101,110,101
  110 PRINT 111
  111 FORMAT(5HKER=2)
  101 CALL PROD(HT,TV2,KN,KP,KP,TV1)
      CALL PROD(P,TV1,KN,KN,KP,G)
      CALL PROD(H,P,KP,KN,KN,TV1)
      CALL PROD(G,TV1,KN,KP,KN,TV2)
      DO 102 I=1,KN
      DO 102 J=1,KN
  102 TV2(I,J)=-TV2(I,J)
      CALL ADD(P,TV2,KN,KN,TV1)
      CALL PROD(PHI,TV1,KN,KN,KN,TV2)
      CALL TRANS(PHI,KN,KN,TV1)
      CALL PROD(TV2,TV1,KN,KN,KN,PNEW)
      CALL ADD(PNEW,Q,KN,KN,PNEW)
      END

      SUBROUTINE TRANS(A,N,M,C)
      DIMENSION A(12,12),C(12,12)
      DO 153 I = 1,N
      DO 153 J=1,M
  153 C(J,I) = A(I,J)
      END

      SUBROUTINE RECIP(N,EP,A,X,KER)
      DIMENSION A(12,12),X(12,12)
      DO 1 I=1,N
      DO 1 J=1,N
    1 X(I,J)=0.0
      DO 2 K=1,N
    2 X(K,K)=1.0
   10 DO 34 L=1,N
      KP=0
      Z=0.0
      DO 12 K=L,N
      IF(Z-ABSF(A(K,L)))11,12,12
   11 Z=ABSF(A(K,L))
      KP = K
   12 CONTINUE
      IF(L-KP)13,20,20
   13 DO 14 J=L,N
      Z=A(L,J)
      A(L,J)=A(KP,J)
   14 A(KP,J)=Z
      DO 15 J=1,N
      Z=X(L,J)
      X(L,J)=X(KP,J)
```

PROGRAM HYBRID II CONTINUED

```
   15 X(KP,J)=Z
   20 IF(ABSF(A(L,L))-EP)50,50,30
   30 IF(L-N)31,34,34
   31 LP1=L+1
      DO 36 K=LP1,N
      IF(A(K,L))32,36,32
   32 RATIO=A(K,L)/A(L,L)
      DO 33 J=LP1,N
   33 A(K,J)=A(K,J)-RATIO*A(L,J)
      DO 35 J=1,N
   35 X(K,J)=X(K,J)-RATIO*X(L,J)
   36 CONTINUE
   34 CONTINUE
   40 DO 43 I=1,N
      II=N+1-I
      DO 43 J=1,N
      S=0.0
      IF(II-N)41,43,43
   41 IIP1=II+1
      DO 42 K=IIP1,N
   42 S=S+A(II,K)*X(K,J)
   43 X(II,J)=(X(II,J)-S)/A(II,II)
      KER=1
      RETURN
   50 KER=2
      END
      END
```

```
SYSTEM RESPONSE TO DETERMINISTIC INPUTS WITH
VARIABLE FILTER GAINS          H G B HALLAS
 .1414     0.1414
 .1414     0.228
 .1414     0.350
 .1414     0.552
 .1414     0.722
```

APPENDIX III

PROGRAM OPTIMUM CONTROLLER

III - 1 General Description

Program Optimum Controller is a CDC 160 digital computer program written for the control of the simulated hybrid system discussed in Section 11 and shown in block diagram form in Figure 11-1.

This program computes a control for each sampled interval given by

$$U(K) = \underline{A}^T \left[\underline{X}(K) - \underline{DI}(K) \right] \qquad \text{(III-1)}$$

where $\underline{A}^T$ is the feedback gain matrix determined by the Fortran program 'Optcon' and is manually entered.

$\underline{X}(K)$ is column vector describing the states of the plant that are sampled by the ADC once each sample interval.

$\underline{DI}(K)$ is column vector describing the deterministic inputs that are generated on the analog computer and are sampled by the ADC once each sample interval.

The program considers all matrices as square matrices and will handle up to fourth order systems as it stands, and higher order systems with an increase in available storage cells for the variables and temporaries.

The program uses subroutines delay, mulop, matadd, negmat and matmul which are 12 bit variable decimal point routines that are described in Appendix VI.

III - 2 Overflow Provisions

Both product and summation overflow error halts are provided in the matrix subroutines. There is no provision for overriding an overflow, however, the scale factor can be changed in cell 27 to allow larger numbers to be used. See signal condition-

ing, Section 9, for further information on number scaling.

III-3 Internal Timing Control

The setting of the internal timing control is explained in subroutine delay in Appendix VI. For the second order plant and a time delay of one second, a coarse setting of $(0012)_8$ and a fine setting of $(0000)_8$ worked reasonably well. However, better accuracy can be obtained by using external timing control.

III-4 External Timing Control

Figure III-1 shows the operation of the external clock with the digital computer and the ADC unit. The input disable jack of the ADC unit is connected by means of an AND-gate to the input ready line of the CDC 160 computer input cable and provides a means of delaying the CDC 160 computer by an external timing device. To input a digital word from the ADC, the CDC 160 computer sends an input request to the ADC at which time the ADC converts the analog voltage to a digital number on the selected A/D channel. If, however, the input ready is held at ground level by means of an external device, the input ready signal to the computer is delayed until the external device drops the input disable from ground to -3 volts. Upon completion of the conversion process, the ADC sends an input ready signal to the computer after which the computer will input the digital word.

The external clock puts out a pulse at the start of each sample. The length of this pulse can be varied, thereby changing the length of the enabling window for the ADC. This is done by changing the value of a capacitor which controls the switching time of a monostable multivibrator. As a guide a 0.1 micro-farad capacitor produces about a 280 micro-second

pulse width. During each sample cycle, as shown in Figure III-2, the ADC must be enabled long enough to allow all channels to be sampled, and inhibited for the remainder of the cycle. Then the computer performs calculations to obtain the desired control, outputs this control D/A, and is hung up waiting for the next clock pulse.

The enabling window must also be short enough so that the next set of samples is not taken until the sample interval has elapsed. The internal time can be adjusted to protect against this possibility by using subroutine delay. For this second order plant, the coarse control was set to $(0001)_8$ and the fine control to $(0000)_8$.

III-5 Use of Program Optimum Controller

A. Set up the plant to be simulated on the PACE TR20 analog computer similar to Figure 11-1 so that each state is sampled by the ADC. Connect the three remote control connections of the analog computer to those of the ADC so that the master clear switch on the CDC 160 digital computer has control of the hybrid simulation.

B. Turn on the ADC/DAC, and the CDC 168 arithmetic unit.

C. Load the program at $(0000)_8$ and the subroutines at addresses specified by the directory cells.

D. Enter the constants that change with the plant and the order of the plant as designated by an asterisk in the following listing of the program.

E. Set the delay loop or the external clock for the sample interval desired, and run the program from $(0000)_8$. The following listing is for the second order gun train plant of

EXTERNAL CLOCK CONTROL

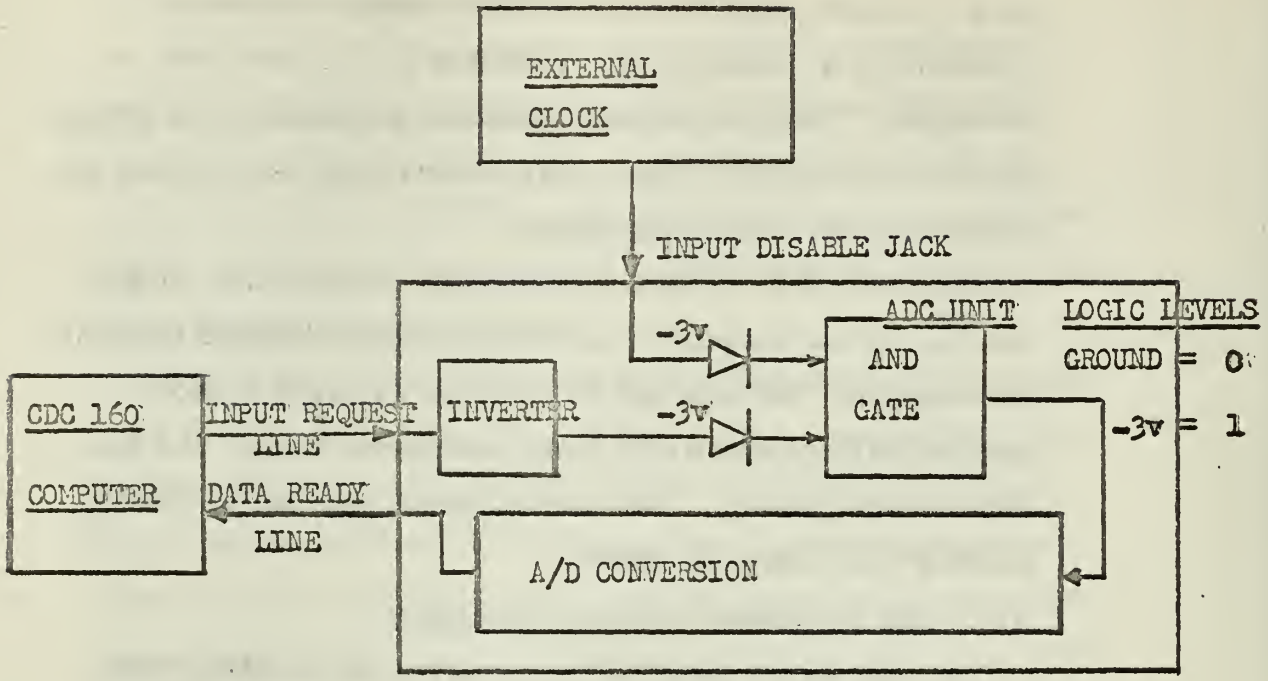


FIGURE III-1

TIME SEQUENCE OF ONE SAMPLE

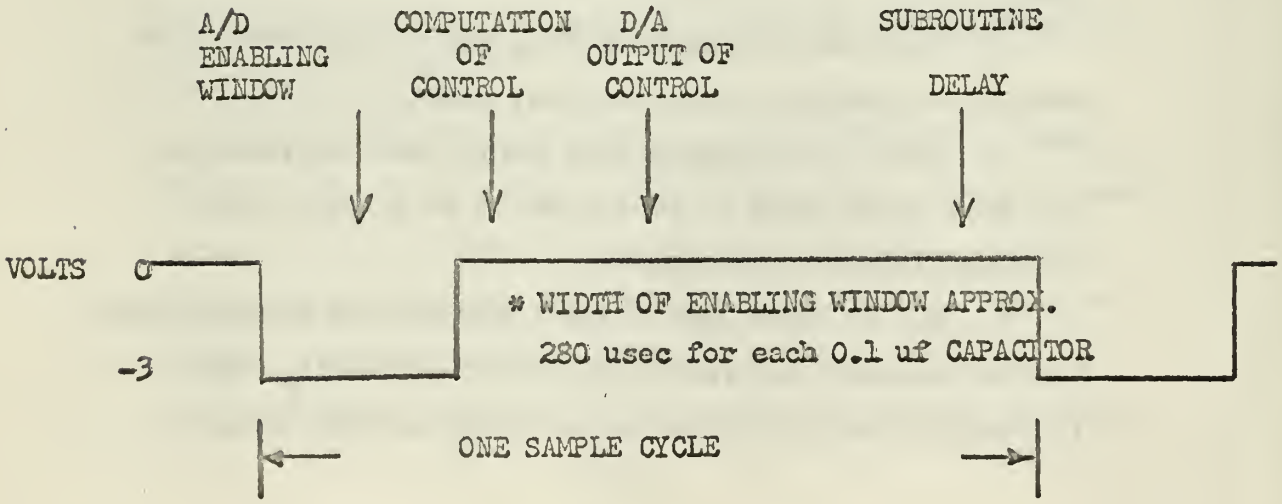


FIGURE III-2

Figure 4-2 with a sample interval of one second provided by an external clock. Three 0.1 micro-farad capacitors in parallel were required to make the width of the enabling window for A/D sampling long enough.

PROGRAM OPTIMUM CONTROLLER

CELL	MACHINE CODE	SYMBOLIC		REMARKS
0000	7101	jfi	1	Run from zero
0001	1000		1000	location of program
				constants to be loaded as noted
0017	0020			counter for null matrix
0024	0004			*product m*n
0026	0000			number of stages, set equal to zero for continuous sampling
0027	0400			scale factor for mulop, scale factor of three
0030	0002			*m matrix dimensions
0031	0002			*n matrix dimensions
0032	0002			*p matrix dimensions
0037	0000			master counter for number of samples
				addresses of subroutines
0052	4300			delay
0053	4400			mulop
0055	4600			matadd
0056	4700			negmat
0057	5000			natmul
				addresses of variables and temporaries
0100 to	0117			null matrix
0120	0000			X(1,1) plant position
0121	0000			
0122	0000			X(2,1) plant velocity
0123	0000			
0140	0000			DI(1,1) position input
0141	0000			
0142	0000			DI(2,1) velocity input
0143	0000			
0160 to	0177			-DI(K)

PROGRAM OPTIMUM CONTROLLER CONTINUED

CELL	MACHINE CODE		SYMBOLIC		REMARKS
0200 to	0217			templ	
0220	7313				*$A^T(1,1)$ position feedback gain
0221	7650				*$A^T(1,2)$ velocity feedback gain
0222	0000				
0223	0000				
0240	0000				U(K) scalar control
0241	0000				
0242	0000				
0243	0000				
1000	2200		ldc		null matrix equal to zero
1001	0100			0100	
1002	4205		stf	a	
1003	2417		lcd	17	-20
1004	4077		std	77	
1005	0400	loopl	ldn	0	
1006	4100		stm		
1007	xxxx	a		xxxx	
1010	5701		aob	a	increment address
1011	5477		aod	77	increment counter
1012	6505		nzb	loopl	
1013	0101		pta		initialize $\underline{X}(K) = 0$
1014	7057		jpi	matmul	
1015	0100			oloo	
1016	0120			0120	
1017	0120			0120	
1020	0101		pta		initialize $\underline{DI}(K) = 0$
1021	7057		jpi	matmul	
1022	0100			0100	
1023	0140			0140	
1024	0140			0140	
1025	0101		pta		initialize $\underline{U}(K) = 0$
1026	7057		jpi	matmul	
1027	0100			0100	
1030	0240			0240	
1031	0240			0240	

PROGRAM OPTIMUM CONTROLLER CONTINUED

CELL	MACHINE CODE		SYMBOLIC		REMARKS
1032	2426		lcd	26	set cell 26 = 0 for continuous sampling
1033	4037		std	37	master counter
1034	2430	loop	lcd	30	input X(K) plant states
1035	4077		std	77	
1036	2200		ldc		
1037	1402			1402	analog to digital channel
1040	4205		stf	b	
1041	2200		ldc		
1042	0120			0120	address of X(K)
1043	4205		stf	c	
1044	7500	loop2	exf		select ADC
1045	xxxx	b		xxxx	
1046	7600		ina		
1047	4100		stm		
1050	xxxx	c		xxxx	
1051	5704		aob	b	increment channel, address & counter
1052	2030		ldd	30	
1053	5303		rab	c	
1054	5477		aod	77	
1055	6511		nzb	loop2	
1056	2430		lcd	30	input DI(K) deterministic inputs
1057	4077		std	77	
1060	2200		ldc		
1061	1407			1407	analog to digital channel
1062	4205		stf	d	
1063	2200		ldc		
1064	0140			0140	address of DI (K)
1065	4205		stf	e	
1066	7500	loop3	exf		select ADC
1067	xxxx	d		xxxx	
1070	7600		ina		
1071	4100		stm		
1072	xxxx	e		xxxx	
1073	5704		aob	d	increment channel, address & counter

PROGRAM OPTIMUM CONTROLLER CONTINUED

CELL	MACHINE CODE	SYMBOLIC		REMARKS
1074	2030	ldd	30	
1075	5303	rab	c	
1076	5477	aod	77	
1077	6511	nzb	loop3	
1100	0101	pta		$\underline{DI}(K) = -\underline{DI}(K)$
1101	7056	jpi	negmat	
1102	0140		$\underline{DI}(K)$	
1103	0160		$-\underline{DI}(K)$	
1104	0101	pta		$templ = \underline{X}(K) - \underline{DI}(K)$
1105	7055	jpi	matadd	
1106	0160		$-\underline{DI}(K)$	
1107	0120		$\underline{X}(K)$	
1110	0200		templ	
1111	0101	pta		$U(K) = \underline{A}^T \left[\underline{X}(K) - DI(K) \right]$ scalar
1112	7057	jpi	matmul	
1113	0220		$\underline{A}^T$	
1114	0200		templ	
1115	0240		U(K)	
1116	7500	exf		output U(K) digital to analog
1117	2411		2411	channel one
1120	7311	out	f	
1121	0241		0241	last word output address plus one
1122	0101	pta		internal time delay
1123	7052	jpi	delay	for external clock control set
1124	0012		coarse	coarse control equal to $(0001)_8$
1125	0000		fine	
1126	5437	aod	37	increment master counter
1127	6573	nzb	loop	
1130	7700	hlt		
1131	0240		0240	first word output address

APPENDIX IV

PROGRAM HYBRID I

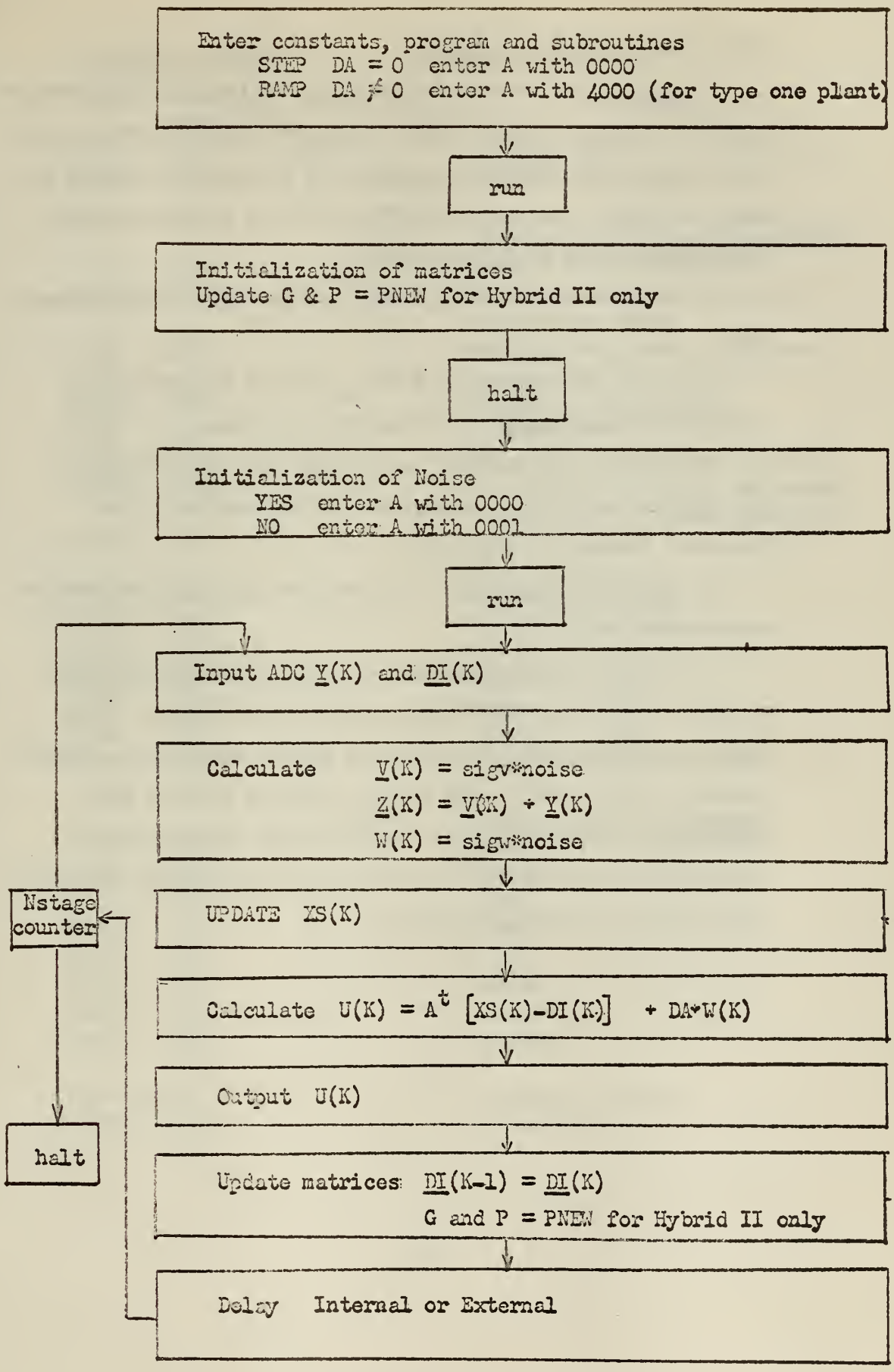
IV-1 General Description

Program Hybrid I is a CDC 160 digital computer program written for the control of the simulated hybrid system discussed in Section 16, shown in block diagram form in Figure 16-1, and in flow chart form in Figure IV-1.

The program uses a filter to estimate all the states of a plant and a controller to synthesize the desired control. The output of the program is a control for the plant simulated on the PACE TR20 analog computer. Stable gain matrices are used for the filter and the controller in Hybrid I. The program considers all matrices as square matrices and will handle up to fourth order systems as it stands, and higher order systems with an increase in available storage cells for the variables and temporaries. The program can be used for plants with more than one observable state. The following listing of the program is for the second order train plant of Figure 4-2 with a sample interval of one second and with a pseudo noise-to-signal power ratio of one taking the noise R(1,1) = 0.02. An enabling window width of 630 microseconds is sufficient to allow all the A/D sampling to be completed on each pass.

The program uses subroutines, xstar, listop, delay, mulop, adop, matadd, negmat and matmul which are 12 bit variable decimal point routines that are described in Appendix VI.

For overflow provisions, internal timing control and external timing control see Sections 2, 3, and 4 of Appendix III for program Optimum Controller.



FLOWCHART FOR HYBRID PROGRAMS

FIGURE IV-1

IV-2 Use of Program Hybrid I

A. Set up the plant to be simulated on the PACE TR20 analog computer similar to Figure 16-1. Connect the three remote control connections of the analog computer to those of the ADC so that the master clear switch on the CDC 160 digital computer has control of the hybrid simulation.

B. Turn on the ADC/DAC unit, and the CDC 168 arithmetic unit.

C. Load the program at $(0000)_8$ and the subroutines at the addresses specified by the directory cells.

D. Enter the constants that change with the plant and the order of the plant as designated by an asterisk in the following listing of the program.

E. Set the delay loop or the external clock for the sample interval desired.

F. Since the gun train plant is a type one system it has a steady state error when responding to a ramp input. A DA factor was introduced to nullify this steady state error. Hence for ramp inputs enter A with $(4000)_8$, run the program from $(0000)_8$ and halt in cell 1047. Enter A with $(0001)_8$ to not reinitialize the noise table otherwise it is restored to the same order, and run the program again.

PROGRAM HYBRID I

CELL	MACHINE CODE	SYMBOLIC		REMARKS
0000	7101	jfi	1	run from zero
0001	1000		1000	location of program
				constants to be loaded as noted
0017	0020			counter for null matrix
0021	0072			*standard deviation of environment noise 'sigw'
0022	0044			*standard deviation of measurement noise 'sigv'
0023	0044			*number of observables
0024	0004			*produce m*n
0025	0000			*DA steady state error correction
0026	0000			[illegible] number of stages, set equal to zero for continuous sampling
0027	0400			scale factor for mulop, scale factor of three
0030	0002			*m matrix dimensions
0031	0002			*n matrix dimensions
0032	0002			*P matrix dimensions
0033	0000			address of W(K)
0035	0000			address of V(K)
0036	0000			address of U(K)
0037	0000			master counter
				addresses of subroutines
0046	4000			xstar
0050	5300			listop
0052	4300			delay
0053	4400			mulop
0054	5200			adop
0055	4600			matadd
0056	4700			negmat
0057	5000			matmul
0060 to 0077				temporary storage
0100 to 0117				null matrix

PROGRAM HYBRID I CONTINUED

CELL	MACHINE CODE	SYMBOLIC	REMARKS
0120	0400		*plant transition matrix PHI(1,1)
0121	0103		PHI(1,2)
0122	0000		PHI(2,1)
0123	0007		PHI(2,2)
0140	0236		*plant distribution matrix DEL(1,1)
0141	0000		
0142	0324		DEL(2,1)
0143	0000		
0160 to 0177			filter gain matrix G
0200	0400		*observable matrix H
0220	7314		*controller gain matrix AT(1,1)
0221	7650		AT(1,2)
0240 to 0257			noisy observable matrix Z(K)
0260 to 0277			filter predicted states XS(K)
0300 to 0317			deterministic input matrix DI(K-1)
0320 to 0337			temp1
0340 to 0357			temp2
0360 to 0377			temp3
0400 to 0417			temp4
0420 to 0437			temp5
0440 to 0457			temp6
0460 to 0477			temp7
0500 to 0517			temp8
0520 to 0537			temp9
0540 to 0557			temp10
0560 to 0577			deterministic input matrix DI(K)
0720	0252		*filter stable gain matrix G(1,1)
0721	0000		
0722	0163		G(2,1)
0723	0000		

PROGRAM HYBRID I CONTINUED

CELL	MACHINE CODE		SYMBOLIC		REMARKS
1000	6205		pjf	a	for ramp input enter A with 4000
1001	2200		ldc		
1002	0452			0452	DA correction for steady state error to ramp input
1003	4025		std	25	
1004	6203		pjf	b	
1005	0400	a	ldn	0	
1006	4025		std	25	DA = 0 for step input
1007	2200	b	ldc		null matrix equal to zero
1010	0100			0100	
1011	4205		stf	c	
1012	2417		lcd	17	-20
1013	4077		std	77	
1014	0400	loopl	ldn	0	
1015	4100		stm		
1016	xxxx	c		xxxx	
1017	5701		aob	c	increment address
1020	5477		aod	77	increment counter
1021	6505		nzb	loopl	
1022	0101		pta		initialize XS(K) = 0
1023	7057		jpi	matmul	
1024	0100			0100	
1025	0260			0260	
1026	0260			0260	
1027	0101		pta		initialize DI(K) = 0
1030	7057		jpi	matmul	
1031	0100			0100	
1032	0560			0560	
1033	0560			0560	
1034	0101		pta		initialize DI(K-1) = 0
1035	7057		jpi	matmul	
1036	0100			0100	
1037	0300			0300	
1040	0300			0300	

PROGRAM HYBRID I CONTINUED

CELL	MACHINE CODE		SYMBOLIC		REMARKS
1041	0101		pta		initialize G = Gstable
1042	7055		jpi	matadd	
1043	0100			0100	
1044	0720			0720	
1045	0160			0160	
1046	0400		ldn	0	
1047	7700		hlt		
					enter A with 0000 noise re-initialized
					enter A with 0001 noise not re-initialized
1050	6120		nzf	d	
1051	2600		lcc		initialization of noise transfers
1052	0144			0144	noise values from 3400-3543 to
1053	4075		std	75	3600-3743 so that they can be
1054	2200		ldc		in the same order for each run
1055	3400			3400	
1056	4077		std	77	
1057	2200		ldc		
1060	3600			3600	
1061	4076		std	76	
1062	2177	loop2	ldi	77	
1063	4176		sti	76	
1064	5477		aod	77	
1065	5476		aod	76	
1066	5475		aod	75	
1067	6560		nzb	loop2	
1070	2426	d	lcd	26	set master counter
1071	4037		std	37	
1072	0101		pta		set program loop link
1073	4060		std	60	
1074	2423		lcd	23	input observables Y(K) by ADC
1075	4077		std	77	

PROGRAM HYBRID I CONTINUED

CELL	MACHINE CODE	SYMBOLIC		REMARKS
1076	2200	ldc		
1077	1402		1402	analog to digital channel
1100	4205	stf	e	
1101	2200	ldc		
1102	0240		0240	address of Y(K)
1103	4205	stf	f	
1104	7500	loop3 exf		select ADC
1105	xxxx	e	xxxx	
1106	7600	ina		
1107	4100	stm		
1110	xxxx	f	xxxx	
1111	5704	aob	e	increment channel, address & counter
1112	2030	ldd	30	
1113	5303	rab	f	
1114	5477	aod	77	
1115	6511	nzb	loop3	
1116	2430	lcd	30	input deterministic signals
1117	4077	std	77	DI(K) by ADC
1120	2200	ldc		
1121	1407		1407	analog to digital channel
1122	4205	stf	g	
1123	2200	ldc		
1124	0560		0560	address of DI(K)
1125	4205	stf	h	
1126	7500	loop4 exf		select ADC
1127	xxxx	g	xxxx	
1130	7600	ina		
1131	4100	stm		
1132	xxxx	h	xxxx	
1133	5704	aob	g	
1134	2030	ldd	30	
1135	5303	rab	h	
1136	5477	aod	77	
1137	6511	nzb	loop4	

PROGRAM HYBRID I CONTINUED

CELL	MACHINE CODE		SYMBOLIC		REMARKS
1140	2423		lcd	23	calculate noisy observable Z(K)
1141	4062		std	62	
1142	2200		ldc		
1143	0240			0240	
1144	4230		stf	i	
1145	4230		stf	j	
1146	0101	loop5	pta		select noise put random number in cell 77
1147	7050		jpi	listop	
1150	4000			4000	
1151	3600			3600	
1152	3743			3743	
1153	2077		ldd	77	preserve random number list
1154	4100		stm		
1155	3743			3743	
					calculate V(K)
1156	2027		ldd	27	scale factor for mulop
1157	4204		stf	k	
1160	0101		pta		
1161	7053		jpi	mulop	
1162	0022			0022	address of sigv
1163	xxxx	k		xxxx	
1164	6102		nzf	l	test mulop overflow
1165	6202		pjf	m	
1166	7700	l	hlt		mulop overflowed, overflow in A register
1167	2077	m	ldd	77	
1170	4035		std	35	V(K) = sigv*random number
1171	0101		pta		Z(K) = Y(K) + V(K)
1172	7054		jpi	adop	
1173	0035			0035	
1174	xxxx	i		xxxx	
1175	xxxx	j		xxxx	
1176	2030		ldd	30	
1177	5303		rab	i	
1200	2030		ldd	30	

PROGRAM HYBRID I CONTINUED

CELL	MACHINE CODE		SYMBOLIC		REMARKS
1201	5304		rab	j	
1212	5462		aod	62	
1203	6535		nzb	loop5	
1204	0101		pta		calculate W(K)
1205	7050		jpi	listop	select noise
1206	4000			4000	put random number in cell 77
1207	3600			3600	
1210	3743			3743	
1211	2077		std	77	preserve random number list
1212	4100		stm		
1213	3743			3743	
1214	2027		ldd	27	scale factor mulop
1215	4204		stf	n	
1216	0101		pta		
1217	7053		jpi	mulop	
1220	0021			0021	address of sigw
1221	xxxx	n		xxxx	
1222	6102		nzf	o	test mulop overflow
1223	6202		pjf	p	
1224	7700	o	hlt		mulop overflowed, overflow in A register
1225	2077	p	ldd	77	
1226	4033		std	33	W(K) = sigw*random number
1227	0101		pta		DI(K) = -DI(K)
1230	7056		jpi	negmat	
1231	0560			0560	
1232	0560			0560	
1233	0101		pta		update XS(K)
1234	7046		jpi	xstar	
1235	0101		pta		temp1 = XS(K) - DI(K)
1236	7055		jpi	matadd	
1237	0260			0260	
1240	0560			0560	
1241	0320			0320	

PROGRAM HYBRID I CONTINUED

CELL	MACHINE CODE	SYMBOLIC		REMARKS
1242	0101	pta		$U(K) = \underline{A}^T\left[\underline{XS}(K) - \underline{DI}(K)\right]$
1243	7057	jpi	matmul	
1244	0220		$\underline{A}^T$	
1245	0320		0320	
1246	0320		0320	
1247	2100	ldm		
1250	0320		0320	
1251	4036		4036	address of U(K)
1252	0101	pta		U(K) = U(K) + DA
1253	7054	jpi	adop	
1254	0036		0036	
1255	0025		0025	
1256	0036		0036	
1257	0101	pta		U(K) = U(K) + W(K)
1260	7054	jpi	adop	
1261	0036		0036	
1262	0033		0033	
1263	0036		0036	
1264	7500	exf		output U(K) by DAC channel one
1265	2411		2411	
1266	7320	out	q	
1267	0037		0037	last word output address plus one
1270	0101	pta		update DI(K-1)
1271	7055	jpi	matadd	
1272	0100		0100	
1273	0560		0560	
1274	0300		0300	
1275	0101	pta		internal time delay
1276	7052	jpi	delay	for external clock control set
1277	0011		coarse	coarse control equal to 0001
1300	0000		fine	

PROGRAM HYBRID I CONTINUED

CELL	MACHINE CODE		SYMBOLIC		REMARKS
1301	5437		aod	37	increment master counter
1302	6102		nzf	r	
1303	6202		pjf	s	
1304	7060	r	jpi	60	return to input of Y(K)
1305	7700	s	hlt		
1306	0036	q		0036	first word output address

Note! If the external clock control is used, the enabling interval for ADC sampling must be long enough to allow all the samples to be made on each pass.

APPENDIX V

PROGRAM HYBRID II

V-1 General Description

Program Hybrid II is a CDC computer program wirtten for the control of the simulated hybrid system discussed in Section 16,' shown in block diagram form in Figure 16-1., and in flow chart form in Figure IV-1.

The program uses a filter to estimate all the states of a plant and a controller to synthesize the desired control. The output of the program is a control for the plant simulated on the PAC TR20 analog computer. A stable gain matrix is used for the controller but a variable gain matrix is computed on line for the filter. The program considers all matrices as square matrices and will handle up to fourth order systems with minor modification beginning at cell 2060, and higher order systems with an increase in available storage cells for the variables and temporaries. The program can be used for plants with only one observable state since a matrix inversion process is replaced by division in the calculation of the filter gain matrix in subroutine vfgain. The following listing of the program is for the second order train plant of Figure 4-2 with a sample interval of one second and with a pseudo noise to signal power ratio of one taking the noise R(1,1) = 0.02. An enabling window width of 600 microseconds is sufficient to allow all the A/D sampling to be completed on each pass.

The program uses subroutines, xstar, listop, delay, mulop, adop, matadd, negmat, matmul, divop and vfgain, which are 12 bit variable decimal point routines that are described in Appendix VI.

For overflow provisions, internal timing control and external timing control see Sections 2, 3 and 4 of Appendix III for program Optimum Controller.

V-2 Use of Program Hybrid II

A. Set up the plant to be simulated on the PACE TR20 analog computer similar to Figure 16-1. Connect the three remote control connections of the analog computer to those of the ADC so that the master clear switch on the CDC 160 digital computer has control of the hybrid simulation.

B. Turn on the ADC/DAC unit, and the CDC 168 arithmetic unit.

C. Load the program at $(0000)_8$ and the subroutines at the addresses specified by the directory cells.

D. Enter constants that change with the plant and the order of the plant as designated by an asterisk in the following listing of the program. For a second order plant enter the initial values of P(1,1) in cell 2061 and P(2,2) in cell 2065. For third or higher order plants rewrite the section beginning at cell 2060 for the initialization of the P matrix.

E. Set the delay loop or external clock for the sample interval desired.

F. Since the gun train plant is a type one system it has a steady state error when responding to a ramp input. A DA factor was introduced to nullify this steady state error. Hence for ramp inputs enter A with $(4000)_8$, run the program from $(0000)_8$ and halt in cell 1047. Enter A with $(0001)_8$ to not reinitialize the noise table; otherwise it is restored to the same order, and run the program again. In subroutine vfgain to provide for a division overflow the variable gain matrix elements are set equal to the

stable gain matrix elements and an error flag indicating the last count is set in cell 16.

PROGRAM HYBRID II

CELL	MACHINE CODE	SYMBOLIC		REMARKS
0000	7101	jfi	1	run from zero
0001	2000		2000	location of program
				constants to be loaded as noted
0017	0020			counter for null matrix
0021	0072			*standard deviation of environment noise 'sigw'
0022	0044			*standard deviation of measurement noise 'sigv'
0023	0001			*number of observables
0024	0004			*produce m*n
0025	0000			*DA steady state error correction
0026	0000			number of stages, set equal to zero for continuous sampling
0027	0400			scale factor for mulop, scale factor of three
0030	0002			*m matrix dimensions
0031	0002			*n matrix dimensions
0032	0002			*p matrix dimensions
0033	0000			address of W(K)
0034	0015			address of sigwsq- variance of measurement noise
0035	0000			address of V(K)
0036	0000			address of U(K)
0037	0000			master counter
				addresses of subroutines
0044	5400			vfgain
0045	4200			divop
0046	4000			xstar
0050	5300			listop
0052	4300			delay
0053	4400			mulop
0054	5200			adop
0055	4600			matadd
0056	4700			negmat
0057	5000			matmul

PROGRAM HYBRID II CONTINUED

CELL	MACHINE CODE	SYMBOLIC	REMARKS
0060 to	0077		temporary storage
0100 to	0117		null matrix
0120	0400		*plant transition matrix PHI(1,1)
0121	0103		PHI(1,2)
0122	0000		PHI(2,1)
0123	0007		PHI(2,2)
0140	0236		*plant distribution matrix DEL(1,1)
0141	0000		
0142	0324		DEL(2,1)
0143	0000		
0160 to	0177		filter gain matrix G
0200	0400		*observable matrix H
0201	0000		
0202	0000		
0203	0000		
0220	7314		*controller gain matrix AT(1,1)
0221	7650		AT(1,2)
0222	0000		
0223	0000		
0240 to	0257		noisy observable matrix Z(K)
0260 to	0277		filter predicted states XS(K)
0300 to	0317		deterministic input matrix DI(K-1)
0320 to	0337		temp1
0340 to	0357		temp2
0360 to	0377		temp3
0400 to	0417		temp4
0420 to	0437		temp5
0440 to	0457		temp6
0460 to	0477		temp7
0500 to	0517		temp8
0520 to	0537		temp9
0540 to	0557		temp10
0560 to	0577		deterministic input matrix DI(K)

PROGRAM HYBRID II CONTINUED

CELL	MACHINE CODE		SYMBOLIC		REMARKS
0600	0005		*environment noise covariance matrix		Q(1,1)
0601	0007				Q(1,2)
0602	0007				Q(2,1)
0603	0011				Q(2,2)
0620 to 0637			best estimate of error covariance matrix P		
0640	0400		*transpose of plant transition matrix		PHIT(1,1)
0641	0000				PHIT(1,2)
0642	0103				PHIT(2,1)
0643	0007				PHIT(2,2)
0660	0005		*measurement noise covariance matrix		R(1,1)
0700 to 0717			PNEW		
0720	0252		*filter stable gain matrix		G(1,1)
0721	0000				
0722	0163				G(2,1)
0723	0000				
2000	6205		pjf	a	for ramp enter A with 4000
2001	2200		ldc		
2002	0452			0452	DA correction for steady state error
2003	4025		std	25	to ramp input
2004	6203		pjf	b	
2005	0400	a	ldn	0	
2006	4025		std	25	DA = 0 for step
2007	2200	b	ldc		set null matrix equal to zero
2010	0100			0100	
2011	4205		stf	c	
2012	2417		lcd	17	-20
2013	4077		std	77	
2014	0400	loop1	ldn	0	
2015	4100		stm		
2016	xxxx	c		xxxx	
2017	5701		aob	c	increment address and counter
2020	5477		aod	77	
2021	6505		nzb	loop1	

PROGRAM HYBRID II CONTINUED

CELL	MACHINE CODE	SYMBOLIC		REMARKS
2022	0101	pta		initialize XS(K) = 0
2023	7057	jpi	matmul	
2024	0100		0100	
2025	0260		0260	
2026	0260		0260	
2027	0101	pta		initialize DI(K) = 0
2030	7057	jpi	matmul	
2031	0100		0100	
2032	0560		0560	
2033	0560		0560	
2034	0101	pta		initialize DI(K-1) = 0
2035	7057	jpi	matmul	
2036	0100		0100	
2037	0300		0300	
2040	0300		0300	
2041	0101	pta		initialize G(K) = 0
2042	7057	jpi	matmul	
2043	0100		0100	
2044	0160		0160	
2045	0160		0160	
2046	0101	pta		initialize PNEW = 0
2047	7057	jpi	matmul	
2050	0100		0100	
2051	0700		0700	
2052	0700		0700	
2053	0101	pta		initialize P(K) = 0
2054	7057	jpi	matmul	
2055	0100		0100	
2056	0620		0620	
2057	0620		0620	
2060	2200	ldc		initialize P(0) values must be inserted

PROGRAM HYBRID II CONTINUED

CELL	MACHINE CODE	SYMBOLIC		REMARKS
2061	0005		0005	*P(1,1)
2062	4100	stm		
2063	0620		0620	
2064	2200	ldc		
2065	0400		0400	*P(2,2)
2066	4100	stm		
2067	0623		0623	Note! this part must be revised for third or higher order plants
2070	0101	pta		update filter gain matrix G
2071	7044	jpi	vfgain	
2072	0101	pta		update P matrix equal to PNEW
2073	7055	jpi	matadd	
2074	0100		0100	
2075	0700		0700	
2076	0620		0620	
2077	0400	ldn	0	
2100	7700	hlt		enter A with 0000 noise reinitialized enter A with 0001 noise not reinitialized
2101	6120	nzf	d	initialization of noise transfers
2102	2600	lcc		noise table from 3400-3543 to
2103	0144		0144	3600-3743 so that they can be in
2104	4075	std	75	the same order for each run
2105	2200	ldc		
2106	3400		3400	
2107	4077	std	77	
2110	2200	ldc		
2111	3600		3600	
2112	4076	std	76	
2113	2177	loop2 ldi	77	
2114	4176	sti	76	
2115	5477	aod	77	
2116	5476	aod	76	
2117	5475	aod	75	
2120	6505	nzb	loop2	

PROGRAM HYBRID II CONTINUED

CELL	MACHINE CODE		SYMBOLIC		REMARKS
2121	2426	d	lcd	26	set master counter
2122	4037		std	37	
2123	0101		pta		set program loop link
2124	4060		std	60	
2125	7500		exf		input observable Y(K) by ADC
2126	1402			1402	ADC channel two
2127	7600		ina		
2130	4100		stm		
2131	0240			0240	address of Y(K)
2132	2430		lcd	30	input deterministic signals DI(K)
2133	4077		std	77	by ADC
2134	2200		ldc		
2135	1407			1407	
2136	4205		stf	e	
2137	2200		ldc		
2140	0560			0560	address of DI(K)
2141	4205		stf	f	
2142	7500	loop3	exf		select ADC
2143	xxxx	e		xxxx	
2144	7600		ina		
2145	4100		stm		
2146	xxxx	f		xxxx	
2147	5704		aob	e	increment channel, address & counter
2150	2030		ldd	30	
2151	5303		rab	f	
2152	5477		aod	77	
2153	6511		nzb	loop3	
2154	0101		pta		calculate noisy observable Z(K)
2155	7050		jpi	listop	
2156	4000			4000	select noise, put noise in cell 77
2157	3600			3600	
2160	3743			3743	
2161	2077		ldd	77	preserve random number list

PROGRAM HYBRID II CONTINUED

CELL	MACHINE CODE		SYMBOLIC		REMARKS
2162	4100		stm		
2163	3743			3743	
					calculate V(K)
2164	2027		ldd	27	scale factor for mulop
2165	4204		stf	g	
2166	0101		pta		
2167	7053		jpi	mulop	
2170	0022			0022	address of sigv
2171	xxxx	g		xxxx	
2172	6102		nzf	h	test mulop overflow
2173	6002		zjf	i	
2174	7700	h	hlt		mulop overflowed, overflow in A register
2175	2077	i	ldd	77	
2176	4035		std	35	V(K) = sigv*random number
2177	0101		pta		Z(K) = Y(K) + V(K)
2200	7054		jpi	adop	
2201	0035			0035	
2202	0240			0240	
2203	0240			0240	
2204	0101		pta		calculate W(K)
2205	7050		jpi	listop	select noise, put noise in cell 77
2206	4000			4000	
2207	3600			3600	
2210	3743			3743	
2211	2077		ldd	77	preserve random number list
2212	4100		stm		
2213	3743			3743	
2214	2027		ldd	27	scale factor for mulop
2215	4204		stf	j	
2216	0101		pta		
2217	7053		jpi	mulop	
2220	0021			0021	address of sigw
2221	xxxx	j		xxxx	
2222	6102		nzf	k	test mulop overflow
2223	6202		pjf	l	

PROGRAM HYBRID II CONTINUED

CELL	MACHINE CODE	SYMBOLIC		REMARKS
2224	7700 k	hlt		mulop overflowed, overflow in A register
2225	2077 l	ldd	77	
2226	4033	std	33	W(K) = sigw*random number
2227	0101	pta		DI(K) = -DI(K)
2230	7056	jpi	negmat	
2231	0560		0560	
2232	0560		0560	
2233	0101	pta		update XS(K)
2234	7046	jpi	xstar	
2235	0101	pta		templ = [XS(K) - DI(K)]
2236	7055	jpi	matadd	
2237	0260		0260	
2240	0560		0560	
2241	0320		0320	
2242	0101	pta		U(K) = $\underline{A}^T$ [XS(K) - DI(K)] scalar
2243	7057	jpi	matmul	
2244	0220		0220	
2245	0320		0320	
2246	0320		0320	
2247	2100	ldm		
2250	0320		0320	
2251	4036	std	36	
2252	0101	pta		U(K) = U(K) + DA
2253	7054	jpi	adop	
2254	0036		0036	
2255	0025		0025	
2256	0036		0036	
2257	0101	pta		U(K) = U(K) + W(K)
2260	7054	jpi	adop	
2261	0036		0036	
2262	0033		0033	
2263	0036		0036	

PROGRAM HYBRID II CONTINUED

CELL	MACHINE CODE		SYMBOLIC		REMARKS
2264	7500		exf		output U(K) by DAC channel one
2265	2411			2411	
2266	7327		out	m	
2267	0037			0037	last word output address plus one
2270	0101		pta		update DI(K-1) = DI(K)
2271	7055		jpi	matadd	
2272	0100			0100	
2273	0560			0560	
2274	0300			0300	
2275	0101		pta		update filter gain matrix G
2276	7044		jpi	vfgain	
2277	0101		pta		update P matrix equal to PNEW
2300	7055		jpi	matadd	
2301	0100			0100	
2302	0700			0700	
2303	0620			0620	
2304	0101		pta		internal time delay
2305	7052		jpi	delay	for external clock control set
2306	0011			coarse	coarse control equal to 0001,
2307	7200			fine	fine to 0000
2310	5437		aod	37	increment master counter
2311	6102		nzf	n	
2312	6202		pjf	o	
2313	7060	n	jpi	60	return to input of Y(K)
2314	7700	o	hlt		
2315	0036	m		0036	first word of output address

Note! If the external clock control is used, the enabling interval for ADC sampling must be long enough to allow all the samples to be made on each pass.

APPENDIX VI

SUBROUTINES FOR CDC 160 DIGITAL PROGRAMS

VI-1 General Description

The subroutines described in this appendix handle decimal numbers up to twelve bits in length and the format of scaling the number is compatible with the CDC 160 computer, the CDC 168 arithmetic unit and the ADC/DAC conversion unit.

The following subroutines were used in programs Hybrid I and II:

Memory allocation	Subroutine
3400 to 3543	noise
4000 to 4107	xstar
4140 to 4171	scalop
4200 to 4220	divop
4300 to 4323	delay
4400 to 4510	mulop
4600 to 4654	matadd
4700 to 4730	negmat
5000 to 5131	matmul
5200 to 5252	adop
5300 to 5370	listop
5400 to 5531	vfgain

All the subroutines, except mulop and listop which were on the library tape, were written to implement programs Optimum Controller, Hybrid I and II. Detailed descriptions of their purpose, calling sequence, and listing are given on the following pages.

VI-2 Mulop

Subroutine mulop uses the CDC 168 arithmetic unit to multiply two twelve bit numbers together and postshift the product. The A register is zero on exit for no overflow and is full positive or negative on exit depending upon the type of overflow.

Overflow can be checked on exit by a non zero jump error halt. Mulop uses temporary cells 77 through 74 among which cell 77 is considered the A register of an arithmetic unit and cell 76, the Q register. The execution time is about 400 microseconds for no scaling and about 550 microseconds with scaling.

The calling sequence is:

```
        std     77          enter multiplier
        pta
        jpi     mulop
                address of multiplicand
                scale factor
        ldd     77          load product
```

VI-3 Listop

Subroutine listop shifts a twelve bit word each time it is called. It can be used to shift data in a table in an end off or circular mode. Listop uses temporary cells 77 through 75. Cell 77 contains the number that is being shifted.

The calling sequence is:

```
        pta
        jpi     listop
                argl        bit 0:  0 - shift down  bit 11:0 - end off
                                    1 - shift up            1-circular
                address of first
                address of last
```

NOISE

Noise is a list of the first 100 random numbers from CDC 1604 library subroutine 'RNDEV'. It has a Gaussian distribution with essentially zero mean, standard deviation of one and a variance of one. It is stored in cells 3400 through 3543. Provision is made in programs so that reinitialization of noise list is at discretion of user. The following columns are a list of the decimal numbers and the octal numbers scaled by a factor of three.

DECIMAL	OCTAL	DECIMAL	OCTAL	DECIMAL	OCTAL
-0.308	7660	-0.654	7530	-0.316	7656
0.963	0366	-1.109	7340	-0.076	7754
0.379	0141	0.207	0065	-0.769	7472
0.379	0521	-0.095	7717	-0.507	7575
-0.006	7776	-0.009	7775	-0.840	7150
-0.079	7753	-0.312	7660	2.172	1054
1.120	0436	-0.725	7506	0.940	0360
-1.232	7304	-0.350	7646	0.773	0306
-0.949	7414	-0.073	7755	0.514	0203
-0.595	7547	-0.853	7445	1.281	0510
-1.046	7364	0.615	0235	-0.856	7444
-0.682	7525	1.132	0537	0.174	0054
-0.750	7477	0.108	0033	-0.827	7454
0.421	0153	0.842	0327	-1.016	7373
-0.756	7476	1.442	0561	0.266	0104
1.969	0770	1.980	0776	0.669	0253
-0.167	7725	1.367	0536	-0.025	7771
2.127	1040	2.605	1232	-0.072	7755
-0.991	7402	-2.013	6774	-1.236	7303
0.408	0150	0.126	0040	-2.245	6701
2.091	1027	0.103	0062	0.717	0267
-0.761	7475	0.852	0662	1.284	0510
-0.951	7414	-1.507	7575	1.252	0500
-0.531	7564	-0.960	7412	-2.360	6643
-0.156	7727	-1.352	7215	0.812	0317
1.075	0423	-0.751	7477	0.543	0213
-0.380	7636	-0.171	7720	-0.491	7602
-0.463	7611	-1.138	7334	-0.081	7723
1.075	0423	-0.771	7476	0.423	0154

DECIMAL	OCTAL	DECIMAL	OCTAL	DECIMAL	OCTAL
-0.380	7636	0.043	0012	-1.891	7033
-0.463	7611	-0.386	7634	-0.082	7752
1.075	0423	-0.814	7457	1.045	0417
-0.476	7604	0.491	0175	-0.401	7631
0.425	0154	-1.992	7001		
0.019	0005				
0.241	0075				

SUBROUTINE XSTAR

Subroutine Xstar solves the controlled recursive filter equation XS(K) = (PHI*XS(K-1) + DEL [A^T [XS(K-1)-DI(K-1)]+DA][1-G*H]+G*Z and updates the value of XS(K) each time it is called.

CALLING SEQUENCE

```
pta
jpi     Xstar
```

CELL	MACHINE CODE	SYMBOLIC		REMARKS
4000	0602	adn	2	return link address
4001	4061	std	60	
4002	0101	pta		
4003	7057	jpi	matmul	
4004	0120		PHI	
4005	260		XS(K-1)	
4006	320		temp1	
4007	0101	pta		
4010	7057	jpi	matmul	
4011	0160		G	
4012	0200		H	
4013	0340		temp2	
4014	0101	pta		
4015	7056	jpi	negmat	
4016	0340		temp2	
4017	0340		-temp2	
4020	0101	pta		
4021	7057	jpi	matmul	
4022	0340		temp2	
4023	0320		temp1	
4024	0360		temp3	temp3 = -G*H*PHI*XS(K-1)
4025	0101	pta		
4026	7055	jpi	matadd	
4027	0320		temp1	
4030	0360		temp3	
4031	0400		temp4	temp4 = [1-G*H]PHI*XS(K-1)
4032	0101	pta		
4033	7055	jpi	matadd	
4034	0260		XS(K-1)	
4035	0300		DI(K-1)	

SUBROUTINE XSTAR CONTINUED

CELL	MACHINE CODE	SYMBOLIC		REMARKS
4036	0420		temp5	temp5 = XS(K-1)-DI(K-1)
4037	0101	pta		
4040	7057	jpi	matmul	
4041	0220		A^T	
4042	0420		temp5	
4043	0540		temp10	temp10 = A^T [XS(K-1)-DI(K-1)]
4044	0101	pta		
4045	7054	jpi	adop	
4046	0540		temp10	
4047	0025		DA	
4050	0540		temp10	
4051	0101	pta		
4052	7057	jpi	matmul	
4053	0140		DEL	
4054	0540		temp10	
4055	0420		temp5	temp5 = DEL[A^T[XS(K-1)-DI(K-1)]+DA]
4056	0101	pta		
4057	7057	jpi	matmul	
4060	0340		temp2	
4061	0420		temp5	
4062	0440		temp6	
4063	0101	pta		
4064	7055	jpi	matadd	
4065	0420		temp5	
4066	0440		temp6	
4067	0460		temp7	temp7 = DEL [A^T[XS(K-1)-DI(K-1)]
4070	0101	pta		+DA] * [1-G*H]
4071	7057	jpi	matmul	
4072	0160		G	
4073	0240		Z	
4074	0500		temp8	temp8 = G*Z
4075	0101	pta		
4076	7055	jpi	matadd	
4077	0400		temp4	
4100	0460		temp7	
4101	0260		XS(K)	XS(K)=(PHI*XS(K-1)+DEL[A^T[XS(K-1)-
4102	0101	pta		DI(K-1)]+DA][1-G*H]
4103	7055	jpi	matadd	
4104	0260		XS(K)	
4105	0500		temp8	
4106	0260		XS(K)	XS(K) = XS(K) + G*Z
4107	7061	jpi	60	return link

SUBROUTINE SCALOP

Subroutine Scalop shifts the binary point of a number, thereby allowing for scaling of octal numbers in the computer on line and exits with the result in the A register. In scaling a number down the binary point is shifted to the right, however, only left shift instructions are available. Hence choose the appropriate number of left shifts from the scaling down table. A variable mask depending upon the type and number of shifts is also given in the tables. This mask discards the lower bits. Scalop uses temporary cells 77 through 74.

CALLING SEQUENCE

```
      pta
      jpi     scalop
              number to be scaled
              variable mask
              number of left shifts
```

SCALING DOWN TABLE

VARIABLE MASKS	LEFT SHIFTS	RIGHT SHIFTS
1777	13	1
0777	12	2
0377	11	3
0177	10	4
0077	7	5
0037	6	6
0017	5	7
0007	4	10
0003	3	11
0001	2	12

SCALING UP TABLE

VARIABLE MASKS	LEFT SHIFTS
3773	1
3771	2
3770	3
3730	4
3710	5
3700	6
3300	7
3100	10
3000	11
2000	12

CELL	MACHINE CODE	SYMBOLIC		REMARKS
4140	0602	adn	2	
4141	4077	std	77	
4142	2177	ldi	77	

SUBROUTINE SCALOP CONTINUED

CELL	MACHINE CODE	SYMBOLIC		REMARKS
4143	4076	std	76	number to be scaled
4144	5477	aod	77	
4145	2177	ldi	77	
4146	4221	stf	mask	variable mask
4147	5477	aod	77	
4150	2577	lci	77	
4151	4075	std	75	- number of left shifts
4152	5477	aod	77	
4153	4074	std	74	return link
4154	2076	ldd	76	
4155	1200	lpc		
4156	4000		4000	
4157	4077	std	77	sign of number
4160	2076 loop	ldd	76	
4161	0102	ls	1	
4162	4076	std	76	
4163	5475	aod	75	increment counter
4164	6504	nzb	loop	shifting completed
4165	2076	ldd	76	
4166	1200	lpc		logical product with mask
4167	xxxx mask		xxxx	
4170	5077	rad	77	scaled number in A register at exit
4171	7074	jpi	74	return main program

SUBROUTINE DIVOP

Subroutine Divop forms the quotient of two numbers that are equally scaled. Since divop calls subroutine mulop, which forms the product of two numbers, by reversing the arguments for subroutine Mulop, there is no error stop for dividing by zero but there are the error stops for full positive and negative overflow. Divop uses temporary cells 77 through 73, and directory cell 53 for mulop.

CALLING SEQUENCE

```
        std     77      enter dividend
        pta
        jpi     divop
                address of divisor
                scale factor
        ldd     77      quotient
```

CELL	MACHINE CODE	SYMBOLIC		REMARKS
4200	0602	adn	2	
4201	4075	std	75	
4202	2175	ldi	75	address of divisor
4203	4074	std	74	
4204	2174	ldi	74	divisor
4205	4211	stf	divisor	
4206	5475	aod	75	
4207	2175	ldi	75	scale factor
4210	4073	std	73	
4211	5475	aod	75	return link
4212	4206	stf	exit	
4213	0101	pta		
4214	7053	jpi	mulop	
4215	0073		0073	
4216	xxxx	divisor	xxxx	
4217	7101	jfi	1	
4220	xxxx	exit	xxxx	

SUBROUTINE DELAY

Subroutine Delay provides a variable-length internal time delay loop controlled by the setting of fine and coarse control arguments. The fine delay loop operates by adding one to the complement of the fine control setting and non zero jumping back. If the fine control setting is $(0000)_8$, then one is added until number $(7777)_8$ negative zero is reached and then the coarse control setting is incremented. This count-down of looping through the fine control delay and adding one to the complement of the coarse control setting is repeated as many times as indicated by the coarse delay factor.

To accurately adjust the internal time delay loops, connect a brush recorder to the plant control, and vary the fine and coarse delays until the desired sample interval is obtained.

CALLING SEQUENCE

```
      pta
      jpi    delay
             coarse
             fine
```

CELL	MACHINE CODE		SYMBOLIC		REMARKS
4300	0602		adn	2	coarse delay address
4301	4206		stf	a	
4302	0601		adn	1	fine delay address
4303	4207		stf	b	
4304	0601		adn	1	return link address
4305	4214		stf	c	
4306	2500		lcm		coarse delay complement
4307	xxxx	a		xxxx	
4310	4212		stf	d	
4311	2500	g	lcm		fine delay complement
4312	xxxx	b		xxxx	
4313	4210		stf	c	

SUBROUTINE DELAY CONTINUED

CELL	MACHINE CODE		SYMBOLIC		REMARKS
4314	5607	f	aof	c	increment fine delay loop
4315	6501		nzb	f	
4316	5604		aof	d	increment coarse delay loop
4317	6506		nzb	g	
4320	7101		jfi	1	
4321	xxxx	c		xxxx	return address
4322	xxxx	d		xxxx	coarse control
4323	xxxx	e		xxxx	fine control

SUBROUTINE MATADD

Subroutine Matadd takes the sum of two matrices at locations A and B, putting the result in location C, and checking for summation overflow. Matadd uses temporary cells 77 through 70. The dimensions of rows (m) and columns (n) of the matrices are entered in cells 30 and 31, respectively, and the address for subroutine mulop in directory cell 53.

CALLING SEQUENCE

```
    pta
    jpi     matadd
            base address of matrix A
            base address of matrix B
            base address of answer matrix C
              m x n    m x n     m x n
MATADD = ( A )  +  (B)  =  ( C )
```

CELL	MACHINE CODE	SYMBOLIC		REMARKS
4600	0602	adn	2	
4601	4077	std	77	
4602	2177	ldi	77	
4603	4073	std	73	base address of matrix A
4604	5477	aod	77	
4605	2177	ldi	77	
4606	4072	std	72	base address of matrix B
4607	5477	aod	77	
4610	2177	ldi	77	
4611	4071	std	71	base address of matrix C
4612	5477	aod	77	
4613	4241	stf	exit	return link
4614	2030	ldd	30	value of m
4615	4077	std	77	multiplier for mulop
4616	0101	pta		
4617	7053	jpi	mulop	
4620	0031		0031	address of n
4621	0001		0001	multiply integer, result in 77
4622	2477	lcd	77	
4623	4070	std	70	counter = -m*n
4624	2173	loop ldi	73	load A(i,j)
4625	3172	adi	72	add B(i,j)

SUBROUTINE MATADD CONTINUED

CELL	MACHINE CODE		SYMBOLIC		REMARKS
4626	4171		sti	71	store in C(i,j)
4627	2173		ldi	73	test for summation overflow
4630	6202		pjf	a	
4631	6307		njf	b	
4632	2172	a	ldi	72	
4633	6313		njf	c	no overflow, A positive-B negative
4634	2171		ldi	71	
4635	6211		pjf	c	no overflow, A and B positive
4636	0401		ldn	1	
4637	7700		hlt		positive overflow, hlt with 1 in
4640	2172	b	ldi	72	A register
4641	6205		pjf	c	no overflow, A negative-B positive
4642	2171		ldi	71	
4643	6303		njf	c	no overflow, A and B negative
4644	0402		ldn	2	
4645	7700		hlt		negative overflow, hlt with 2 in
4646	5473		aod	73	A register
4647	5472		aod	72	increment matrix addresses and
4650	5471		aod	71	counter
4651	5470		aod	70	
4652	6526		nzb	loop	
4653	7101		jfi	1	
4654	xxxx	exit		xxxx	

SUBROUTINE NEGMAT

Subroutine Negmat takes the negative of matrix at location A and stores the answer at location B. Negmat uses temporary cells 77 through 71. The dimensions of rows (m) and columns (n) are entered into cells 30 and 31 respectively, and the address of subroutine mulop in directory cell 53.

CALLING SEQUENCE

```
        pta
        jpi     negmat
                base address of matrix A
                base address of answer matrix B
                  m x n     m x n
        NEGMAT of ( A ) = - ( A )
```

CELL	MACHINE CODE	SYMBOLIC		REMARKS
4700	0602	adn	2	
4701	4077	std	77	
4702	2177	ldi	77	
4703	4073	std	73	base address of matrix A
4704	5477	aod	77	
4705	2177	ldi	77	
4706	4072	std	72	base address of matrix B
4707	5477	aod	77	
4710	4220	stf	exit	return link
4711	2030	ldd	30	value of m
4712	4077	std	77	multiplier for mulop
4713	0101	pta		
4714	7053	jpi	mulop	
4715	0031		0031	address of n
4716	0001		0001	multiply integer, result in 77
4717	2477	lcd	77	counter = -m*n
4720	4071	std	71	
4721	2573 loop	lci	73	load -A(i,j)
4722	4172	sti	72	store in B(i,j)
4723	5473	aod	73	increment matrix address and
4724	5472	aod	72	counter
4725	5471	aod	71	
4726	6505	nzb	loop	
4727	7101	jfi	1	
4730	xxxx exit		xxxx	

SUBROUTINE MATMUL

Subroutine Matmul forms the produce of matrices in locations A and B, storing the result in location C, and checking for summation and product overflow. Matmul uses temporary cells 77 through 64. The scale factor for mulop is entered in cell 27, the dimensions of the matrices (m, n, p) in cells 30, 31 and 32, and the address of subroutine mulop in directory cell 53.

CALLING SEQUENCE

```
        pta
        jpi     matmul
                base address of matrix A
                base address of matrix B
                base address of answer matrix C
              m x p   p x n   m x n
MATMUL = ( A ) * ( B ) = ( C )
```

CELL	MACHINE CODE	SYMBOLIC		REMARKS
5000	0602	adn	2	
5001	4077	std	77	
5002	2177	ldi	77	
5003	4073	std	73	base address of matrix A
5004	5477	aod	77	
5005	2177	ldi	77	
5006	4072	std	72	base address of matrix B
5007	5477	aod	77	
5010	2177	ldi	77	
5011	4071	std	71	base address of matrix C
5012	5477	aod	77	
5013	4244	stf	exit	
5014	2027	ldd	27	scale factor for mulop
5015	4231	stf	sf	
5016	2032	ldd	32	value of p
5017	4077	std	77	multiplier for mulop
5020	0101	pta		
5021	7053	jpi	mulop	
5022	0031		0031	address of n

SUBROUTINE MATMUL CONTINUED

CELL	MACHINE CODE	SYMBOLIC		REMARKS
5023	0001		0001	multiply integer, answer in 77
5024	2477	lcd	77	-p*n
5025	0601	adn	1	
5026	4064	std	64	counter = 1-p*n
5027	2430	lcd	30	-m
5030	4065	std	65	m counter
5031	2431	loop3 lcd	31	-n
5032	4066	std	66	n counter
5033	0400	loop2 ldn	0	
5034	4070	std	70	set temporary sum equal to zero
5035	2432	lcd	32	-p
5036	4067	std	67	p counter
5037	2173	loop1 ldi	73	value of A(i,k)
5040	4077	std	77	multiplier for mulop
5041	2072	ldd	72	address of B(k,j)
5042	4203	stf	mul	
5043	0101	pta		
5044	7053	jpi	mulop	
5045	xxxx	mul	xxxx	
5046	xxxx	sf	xxxx	
5047	6161	nzf	hlt	mulop overflowed
5050	2077	ldd	77	a(i,k)*B(k,j)
5051	4063	std	63	
5052	6206	pjf	a	check summation overflow
5053	6314	njf	b	i.e. A(1,1)*B(1,1)+
5054	0401	ldn	1	A(1,2)*B(2,1)
5055	6103	nzf	a	absolute jump forward
5056	7101	jfi	1	
5057	xxxx	exit	xxxx	return address
5060	2070	a ldd	70	temporary sum
5061	6315	njf	c	no overflow possible, opposite
5062	2063	ldd	63	signs
5063	5070	rad	70	check temporary sum
5064	6214	pjf	d	
5065	0404	ldn	4	positive summation overflow, hlt
5066	7700	hlt		with 4 in A register
5067	2070	b ldd	70	
5070	6206	pjf	c	no overflow possible, opposite
5071	2063	ldd	63	signs

SUBROUTINE MATMUL CONTINUED

CELL	MACHINE CODE	SYMBOLIC		REMARKS
5072	5070	rad	70	check temporary sum
5073	6305	njf	d	
5074	0405	ldn	5	negative summation overflow, hlt
5075	7700	hlt		with 5 in A register
5076	2063 c	ldd	63	
5077	5070	rad	70	temporary sum
5100	2031 d	ldd	31	increment B(k,j) by n
5101	5072	rad	72	
5102	0401	ldn	1	increment A(i,j) by one
5103	5073	rad	73	
5104	5467	aod	67	loop for p products
5105	6546	nzb	loop1	
5106	2070	ldd	70	temporary sum
5107	4171	sti	71	store in C(i,j)
5110	0401	ldn	1	
5111	5071	rad	71	increment C(i,j)
5112	2432	lcd	32	-p
5113	5073	rad	73	return A(i,k) to base address
5114	2064	ldd	64	
5115	5072	rad	72	return B(k,j) to base address
5116	5466	aod	66	plus one loop for n columns
5117	6564	nzb	loop2	
5120	2032	ldd	32	p
5121	5073	rad	73	advance address of A(i,k) to A(i,k) + 1
5122	2431	lcd	31	-n
5123	5072	rad	72	return B(k,j) to base address
5124	5465	aod	65	loop of m rows
5125	6574	nzb	loop3	
5126	6650	pjb	exit	absolute jump backward to exit
5127	6751	njb	exit	
5130	0403 hlt	ldn	3	hlt with 3 in A register, mulop
5131	7700	hlt		overflowed, check scale factor cell 27

SUBROUTINE ADOP

Subroutine Adop takes the sum of scalars at addresses A and B and puts the result in C, and checks for summation overflow. The A register is zero on exit for no overflow and is full positive or negative on exit depending upon the type of overflow. Overflow can be checked on exit by a non zero jump error halt. Adop uses cells 77 through 73.

CALLING SEQUENCE

```
        pta
        jpi     adop
                address of A
                address of B
                address of C
```

CELL	MACHINE CODE	SYMBOLIC		REMARKS
5200	0602	adn	2	
5201	4077	std	77	
5202	2177	ldi	77	
5203	4076	std	76	address of A
5204	5477	aod	77	
5205	2177	ldi	77	
5206	4075	std	75	address of B
5207	5477	aod	77	
5210	2177	ldi	77	
5211	4074	std	74	address of C
5212	5477	aod	77	
5213	4073	std	73	return link
5214	2176	ldi	76	value of A
5215	6202	pjf	a	
5216	6314	njf	b	
5217	2175 a	ldi	75	value of B
5220	6327	njf	c	no overflow possible, opposite
5221	3176	adi	76	signs
5222	4174	sti	74	C = A + B
5223	6226	pjf	d	no overflow
5224	6303	njf	errorP	
5225	2073	ldd	73	
5226	4020	std	20	

SUBROUTINE ADOP CONTINUED

CELL	MACHINE CODE	SYMBOLIC		REMARKS
5227	2216 errorP	ldf	fullP	
5230	4077	std	77	
5231	6221	pjf	exit	positive summation overflow
5232	2175 b	ldi	75	
5233	6214	pjf	c	
5234	3176	adi	76	
5235	4174	sti	74	C = A + B
5236	6313	njf	d	no overflow possible, opposite
5237	6203	pjf	errorN	signs
5240	2073	ldd	73	
5241	4020	std	20	
5242	2204 errorN	ldf	fullN	
5243	4077	std	77	
5244	6306	njf	exit	negative summation overflow
5245	3777		fullP	
5246	4000		fullN	
5247	3176 c	adi	76	
5250	4174	sti	74	C = A + B
5251	0400 d	ldn	0	
5252	7073 exit	jpi	73	

SUBROUTINE VFGAIN

Subroutine Vfgain solves the following recursive relationships in updating the filter gain matrix G, and the conditional covariance matrix P.

$$G(K) = P(K)*H^T[H*P(K)*H^T + R]^{-1}$$

$$P(K) = PHI*[P(K-1) - G(K-1)*H*P(K-1)]*PHI^T + Q$$

CALLING SEQUENCE

```
pta
jpi     vfgain
```

CELL	MACHINE CODE	SYMBOLIC		REMARKS
5400	0602	adn	2	
5401	4061	std	60	return link
5402	0101	pta		
5403	7057	jpi	matmul	
5404	0620		P	
5405	0200		H^T	
5406	0320		temp1	temp1 = $P*H^T$
5407	0101	pta		
5410	7057	jpi	matmul	
5411	0200		H	
5412	0320		temp1	
5413	0340		temp2	
5414	0101	pta		
5415	7055	jpi	matadd	
5416	0340		temp2	
5417	0660		R	
5420	0360		temp3	temp3 = $H*P*H^T$ + R scalar
5421	2430	lcd	30	-m
5422	4063	std	63	
5423	2200	ldc		since there is only one observ-
5424	0160		0160	able for this plant, matrix
5425	4233	stf	a	inversion can be replaced by
5426	2200	ldc		division to obtain the filter
5427	0320		0320	gain matrix elements
5430	4205	stf	b	
5431	2200	ldc		
5432	0360		0360	

SUBROUTINE VFGAIN CONTINUED

CELL	MACHINE CODE	SYMBOLIC		REMARKS
5433	4210	stf	c	
5434	2100 loop1	ldm		
5435	xxxx b		xxxx	
5436	4077	std	77	dividend for divop
5437	2027	ldd	27	scale factor for divop
5440	4204	stf	sf	
5441	0101	pta		
5442	7045	jpi	divop	
5443	xxxx c		xxxx	$G(1,1)=P(1,1)*H^T/(H*P*H^T+R)$
5444	xxxx sf		xxxx	$G(2,1)=P(2,1)*H^T/(H*P*H^T+R)$
5445	6011	zjf	d	
5446	2037	ldd	37	master counter
5447	4016	std	16	set error flag for divop overflow
5450	0101	pta		
5451	7055	jpi	matadd	set G=G stable for divop overflow
5452	0100		zero	
5453	0720		Gstable	
5454	0160		G	
5455	7061	jpi	61	return main program
5456	2077	ldd	77	quotient from divop
5457	4100	stm		
5460	xxxx a		xxxx	store G matrix elements
5461	2030	ldd	30	
5462	5302	rab	2	increment addresses of G and temp1
5463	2030	ldd	30	
5464	5327	rab	b	
5465	5463	aod	63	increment counter
5467	0101	pta		
5470	7057	jpi	matmul	update P matrix
5471	0200		H	
5472	0620		P	
5473	0400		temp4	
5474	0101	pta		
5475	7057	jpi	matmul	
5476	0160		G	
5477	0400		temp4	
5500	0420		temp5	
5501	0101	pta		

SUBROUTINE VFGAIN CONTINUED

CELL	MACHINE CODE	SYMBOLIC		REMARKS
5502	7056	jpi	negmat	
5503	0420		temp5	
5504	0420		-temp5	temp5 = - G*H*P
5505	0101	pta		
5506	7055	jpi	matadd	
5507	0620		P	
5510	0420		temp5	
5511	0440		temp6	
5512	0101	pta		
5513	7057	jpi	matmul	
5514	0120		PHI	
5515	0440		temp6	
5516	0460		temp7	
5517	0101	pta		
5520	7057	jpi	matmul	
5521	0460		temp7	
5522	0640		PHI^T	
5523	0700		PNEW	$PNEW=PHI(P - G*H*P)PHI^T$
5524	0101	pta		
5525	7055	jpi	matadd	
5526	0700		PNEW	
5527	0600		Q	
5530	0700		PNEW	PNEW = PNEW + Q
5531	7061	jpi	61	return main program

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	20
2.	Library U. S. Naval Postgraduate School Monterey, California 93940	2
3.	Prof. James S. Demetry (Thesis Advisor) Department of Electrical Engineering U. S. Naval Postgraduate School Monterey, California 93940	2
4.	Lt. Henry George Bouchier Hallas 1111 College Street West Toronto 4, Ontario, Canada	1
5.	Librarian Bureau of Naval Weapons Washington, D. C. 20360	1
6.	Commanding Officer and Director U. S. Navy Electronics Laboratory (Library) San Diego 52, California	1

DOCUMENT CONTROL DATA - R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) U. S. Naval Postgraduate School Monterey, California 93940	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

3. REPORT TITLE

The Hybrid Simulation of Tactical Weapon Discrete Filter-Controller

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

Masters' Thesis, May 1966

5. AUTHOR(S) (Last name, first name, initial)

Hallas, Henry G. B., Lieutenant, Royal Canadian Navy

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
May 1966	144	5

8a. CONTRACT OR GRANT NO. b. PROJECT NO. c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) 9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)

10. AVAILABILITY/LIMITATION NOTICES

This document has been approved for public release and sale; its distribution is unlimited.

Qualified requestors may obtain copies of this report from DDC. Mcmeken 10/9/69

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Bureau of Naval Weapons

13. ABSTRACT

This hybrid simulation demonstrates the feasibility of using a digital computer for the realization of a discrete filter-controller to drive a 3"/50 gun mount in response to step, ramp and stabilization orders. The gun position designation signals are subject to random environmental noise and the observation of position is contaminated by random measurement noise. These noise sources are assumed to exhibit a normal distribution with zero mean. A filter is used to predict plant position and velocity from a noisy observable position, and a controller is used to provide the desired feedback of these states for a ripple-free response. The simulation results indicate that there is no detectable difference in response between constant filter gains and adaptive filter gains for this time-invariant plant. The results also demonstrate that the application of good programming techniques is paramount in minimizing the timing and scaling limitations imposed by hybridization.

DD FORM 1 JAN 64 1473

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Digital Controller Digital Filter Discrete filter-controller Hybrid Simulation Sampled-data control						

INSTRUCTIONS

1. ORIGINATING ACTIVITY: Enter the name and address of the contractor, subcontractor, grantee, Department of Defense activity or other organization *(corporate author)* issuing the report.

2a. REPORT SECURITY CLASSIFICATION: Enter the overall security classification of the report. Indicate whether "Restricted Data" is included. Marking is to be in accordance with appropriate security regulations.

2b. GROUP: Automatic downgrading is specified in DoD Directive 5200.10 and Armed Forces Industrial Manual. Enter the group number. Also, when applicable, show that optional markings have been used for Group 3 and Group 4 as authorized.

3. REPORT TITLE: Enter the complete report title in all capital letters. Titles in all cases should be unclassified. If a meaningful title cannot be selected without classification, show title classification in all capitals in parenthesis immediately following the title.

4. DESCRIPTIVE NOTES: If appropriate, enter the type of report, e.g., interim, progress, summary, annual, or final. Give the inclusive dates when a specific reporting period is covered.

5. AUTHOR(S): Enter the name(s) of author(s) as shown on or in the report. Enter last name, first name, middle initial. If military, show rank and branch of service. The name of the principal author is an absolute minimum requirement.

6. REPORT DATE: Enter the date of the report as day, month, year; or month, year. If more than one date appears on the report, use date of publication.

7a. TOTAL NUMBER OF PAGES: The total page count should follow normal pagination procedures, i.e., enter the number of pages containing information.

7b. NUMBER OF REFERENCES: Enter the total number of references cited in the report.

8a. CONTRACT OR GRANT NUMBER: If appropriate, enter the applicable number of the contract or grant under which the report was written.

8b, 8c, & 8d. PROJECT NUMBER: Enter the appropriate military department identification, such as project number, subproject number, system numbers, task number, etc.

9a. ORIGINATOR'S REPORT NUMBER(S): Enter the official report number by which the document will be identified and controlled by the originating activity. This number must be unique to this report.

9b. OTHER REPORT NUMBER(S): If the report has been assigned any other report numbers *(either by the originator or by the sponsor)*, also enter this number(s).

10. AVAILABILITY/LIMITATION NOTICES: Enter any limitations on further dissemination of the report, other than those imposed by security classification, using standard statements such as:

(1) "Qualified requesters may obtain copies of this report from DDC."

(2) "Foreign announcement and dissemination of this report by DDC is not authorized."

(3) "U. S. Government agencies may obtain copies of this report directly from DDC. Other qualified DDC users shall request through _____."

(4) "U. S. military agencies may obtain copies of this report directly from DDC. Other qualified users shall request through _____."

(5) "All distribution of this report is controlled. Qualified DDC users shall request through _____."

If the report has been furnished to the Office of Technical Services, Department of Commerce, for sale to the public, indicate this fact and enter the price, if known.

11. SUPPLEMENTARY NOTES: Use for additional explanatory notes.

12. SPONSORING MILITARY ACTIVITY: Enter the name of the departmental project office or laboratory sponsoring *(paying for)* the research and development. Include address.

13. ABSTRACT: Enter an abstract giving a brief and factual summary of the document indicative of the report, even though it may also appear elsewhere in the body of the technical report. If additional space is required, a continuation sheet shall be attached.

It is highly desirable that the abstract of classified reports be unclassified. Each paragraph of the abstract shall end with an indication of the military security classification of the information in the paragraph, represented as *(TS), (S), (C), or (U)*.

There is no limitation on the length of the abstract. However, the suggested length is from 150 to 225 words.

14. KEY WORDS: Key words are technically meaningful terms or short phrases that characterize a report and may be used as index entries for cataloging the report. Key words must be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location, may be used as key words but will be followed by an indication of technical context. The assignment of links, rules, and weights is optional.